BUY IN PRINT WORLDWIDE MAGPI.CC/STORE

The MagPi

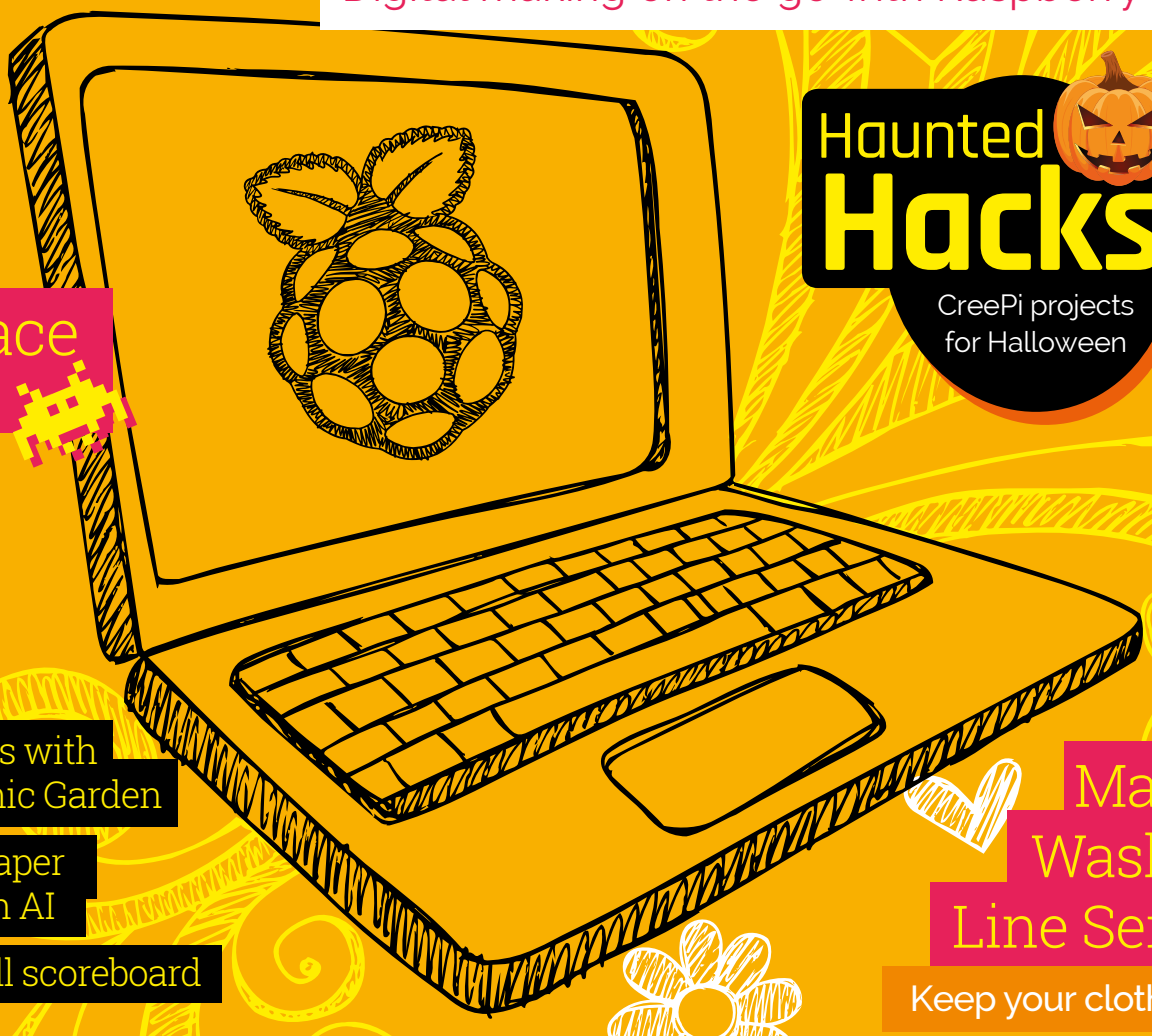
Issue 74 | October 2018 | magpi.cc

The official Raspberry Pi magazine

BUILD A LAPTOP

Digital making on the go with Raspberry Pi

WIN! Media Centre Kits



Haunted Hacks

CreePi projects for Halloween

Code Space Invaders

Recreate this classic in Python

Plus

- Grow veggies with the Aquaponic Garden
- Play Rock, paper scissors with AI
- Table football scoreboard

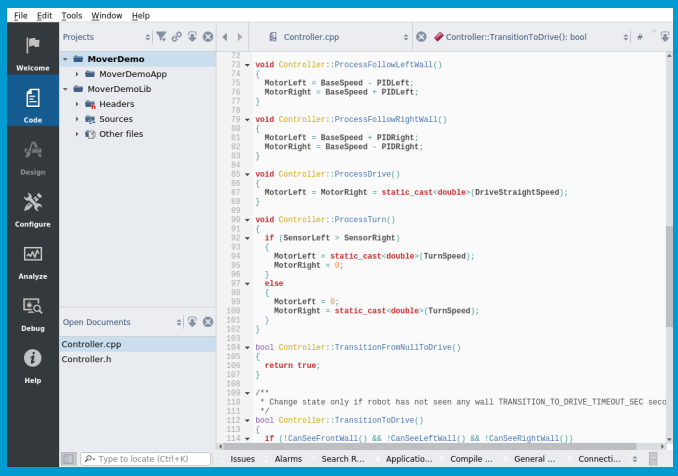
Make a Washing Line Sensor

Keep your clothes dry

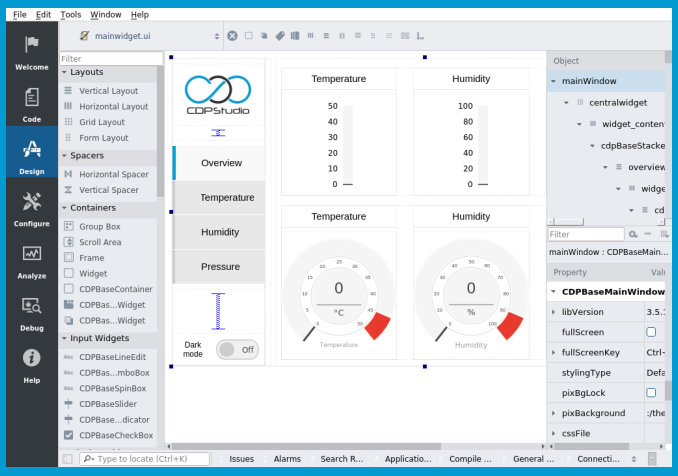
New Picade | Top 10 Cases | QuickStart guide

Raspberry Pi PRESS

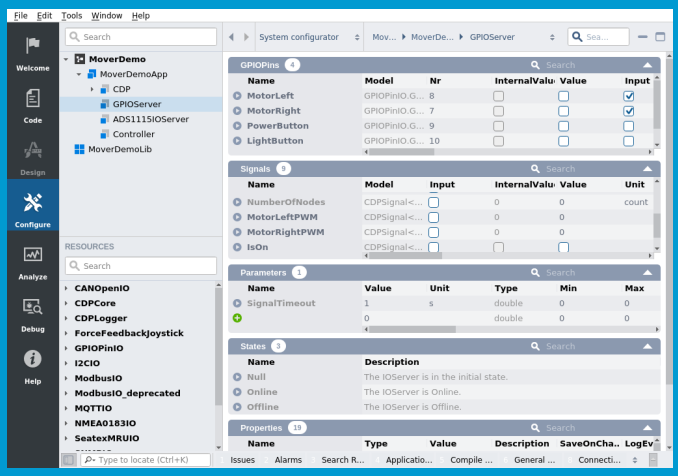
GLOBAL DELIVERY magpi.cc/store



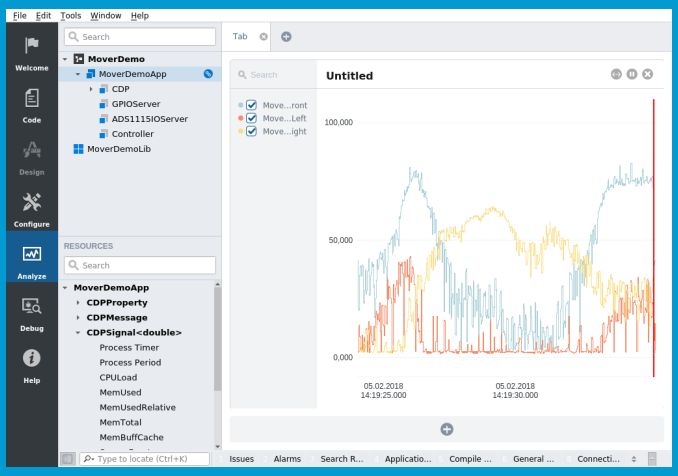
Code



Design



Configure



Analyze

Now free for home projects

A professional control system development tool

CDP Studio is a development platform for industrial control systems, now coming with a free version for non-commercial use. The system can run on a Raspberry Pi, supports C++, open source libraries and has a large feature toolbox including GPIO, I2C and MQTT. Its built in GUI design tool and features lets you code less and do more.

Free download on www.cdpstudio.com

CDP Technologies AS
Nedre Strandgate 29
P.O. Box 144
NO-6001 Ålesund, Norway

Tel: +47 990 80 900
info@cdptech.com
www.cdpstudio.com

CDPStudio

WELCOME
to the new look MagPi

Regular readers will notice a big change this month. We've been working hard on this issue for many months; so, welcome to your amazing new look version of *The MagPi*.

We believe *The MagPi* is the best computer magazine on the newsstand. Every issue is packed with physical computing advice, code, and the most amazing projects from the greatest computer community on Earth.

So how do you make the best computer magazine even better? Throughout this edition, you'll find bigger photographs of brilliant community projects; the latest Raspberry Pi product information; cleaner and clearer fonts for both text and code; and a fresh new design that's brighter than ever. We've made space for bigger features on the most important Raspberry Pi computing projects, tutorials, and reviews.

In short, *The MagPi* is better than ever. We hope you love this new look as much as we do.

Lucy Hattersley Editor

PS: We're 75 issues young next month. Stay tuned!

EDITOR

Lucy Hattersley

Editor of *The MagPi*. Lucy codes, crafts, and creates wonky robots. She speaks French (badly) and mangles the piano. One day she'll get that pet dog.

magpi.cc

GET A PI ZERO W WITH A SUBSCRIPTION! PAGE 30

Contents

Issue 74 > October 2018

Cover Feature

20 Build your own laptop

Regulars

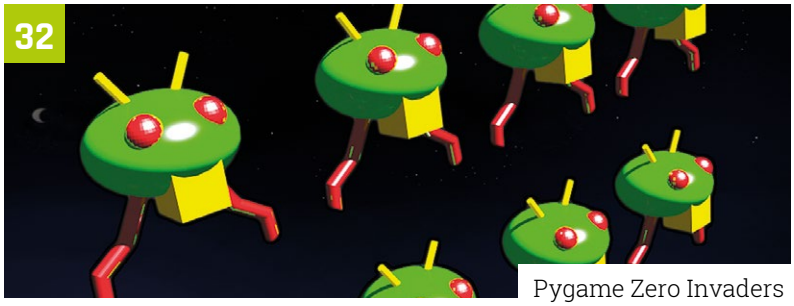
06 News
66 Technical FAQ
94 Your letters
98 Final word

Project Showcases

08 Aquaponic Garden
12 PiCam Marine camera
14 FRILLER robot
16 Table Foosball
18 Rock, Paper, Scissors

PiCam Marine camera

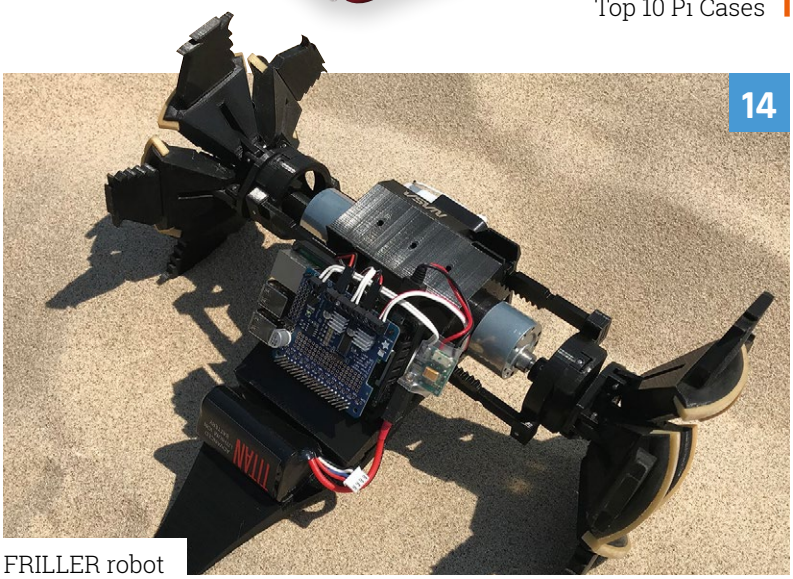
Table Foosball



Pygame Zero Invaders

DISCLAIMER: Some of the tools and techniques shown in The MagPi magazine are dangerous unless used with skill, experience, and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. Children should be supervised. Raspberry Pi (Trading) Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in The MagPi magazine. Laws and regulations covering many of the topics in The MagPi magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in The MagPi magazine may go beyond. It is your responsibility to understand the manufacturer's limits.

80 Top 10 Pi Cases



14 FRILLER robot

78 Kitty Ears

76 Picade

Tutorials

32 Pygame Zero Invaders
40 Pi Bakery: Matrix – part 2
46 Laundry-saving rain detector
50 Minecraft with Wolfram
56 Make games in C – part 10
60 Pi QuickStart guide

The Big Feature

Reviews

76 Picade
78 Kitty Ears
80 10 best Raspberry Pi cases
82 Learn C with Raspberry Pi

Community

86 Interview
88 This month in Raspberry Pi
92 Events

WIN One of Five Pi Switch Caps

with an OLED display and weather pack!

96

In association with NANOMESHER

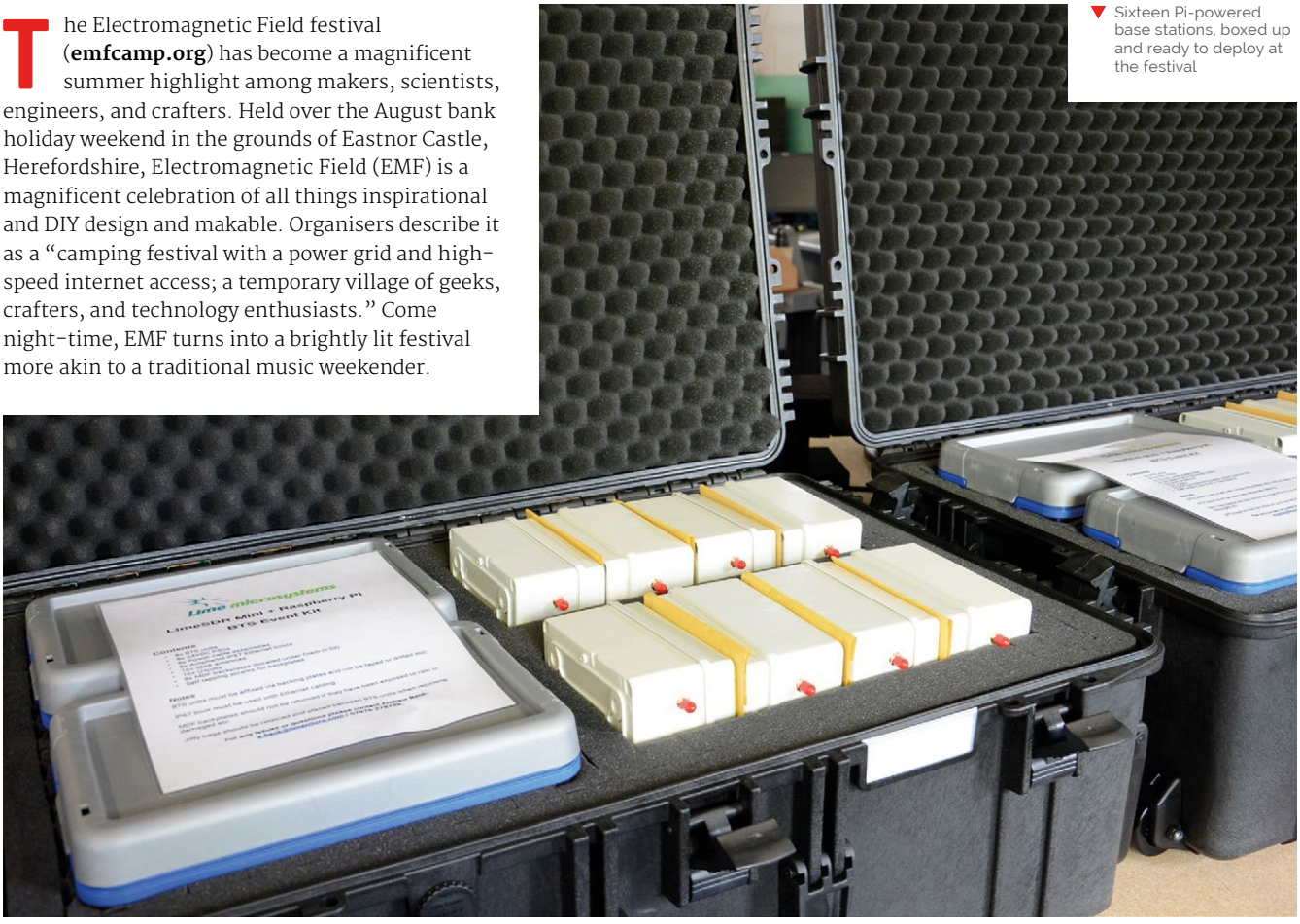
EMF festival powered by Raspberry Pi network

LimeSDR guarantees no rain on the parade of electronic making, reports **Rosie Hattersley**

▲ A LimeSDR mounted base station providing GSM for EMF festival-goers

▼ Sixteen Pi-powered base stations, boxed up and ready to deploy at the festival

The Electromagnetic Field festival (**emfcamp.org**) has become a magnificent summer highlight among makers, scientists, engineers, and crafters. Held over the August bank holiday weekend in the grounds of Eastnor Castle, Herefordshire, Electromagnetic Field (EMF) is a magnificent celebration of all things inspirational and DIY design and makable. Organisers describe it as a "camping festival with a power grid and high-speed internet access; a temporary village of geeks, crafters, and technology enthusiasts." Come night-time, EMF turns into a brightly lit festival more akin to a traditional music weekender.



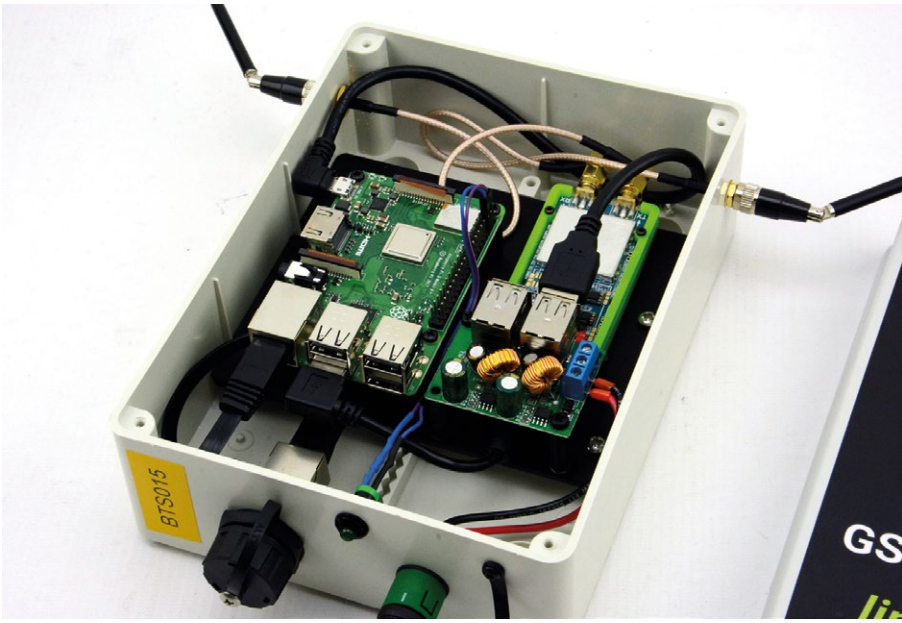
Weatherproofing the boards was essential, given the British weather

Inside the base station is a Raspberry Pi 3B+ and LimeSDR board

But electrical and electronic kit does not obviously lend itself to being assembled and demonstrated in a rural field, particularly when said field is situated in the British countryside. Chancing the great British summer with millions of pounds' worth of people's highly personal tech

> The base stations combined a LimeSDR Mini board with a Raspberry Pi 3 Model B+

at stake is a foolhardy venture. Not only that, but even festival-goers' badges are tech-heavy. Entry depends on a badge that is also a full-blown microcontroller, with WiFi and an LCD screen, that participants can take home and reprogram.



Mobile connectivity

Sensibly enough, the organisers of EMF not only recruited a mobile communications network provider to ensure internet connectivity for festival-goers, but they also weatherproofed the entire shebang. Lime Microsystems (**limemicro.com**) put together a self-contained GSM network and wireless infrastructure consisting of 16 weatherproof base stations. The base stations combined a LimeSDR Mini board with a Raspberry Pi 3 Model B+ host running Linux on a fully open-source GSM base station (BTS) stack and Osmocom cellular network software.

The mounted base stations enjoyed a mains power supply, via a 24 V adapter and DC-DC converter, making it critical that their Raspberry Pi enclosures were fully waterproof and allowed no moisture to ingress. The upshot: the festival's technological mavens could confidently show off their innovations and ideas with no fear of rain stopping play.

MAKER

Diego Braga

An IT DevOps engineer with a passion for automation and IoT projects.

@braghettos

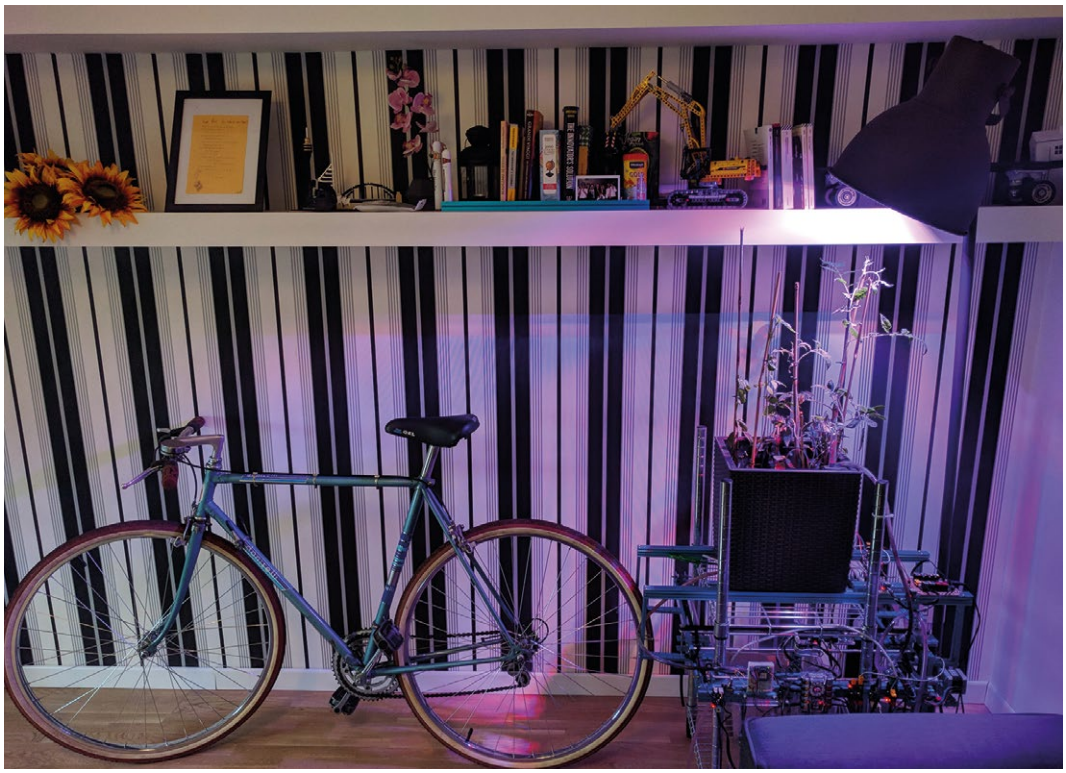
Aquaponic Domestic Garden

Growing veggies indoors when you're not always there is easy when you have a Raspberry Pi and a guard fish. **Rob Zwetsloot** learns more

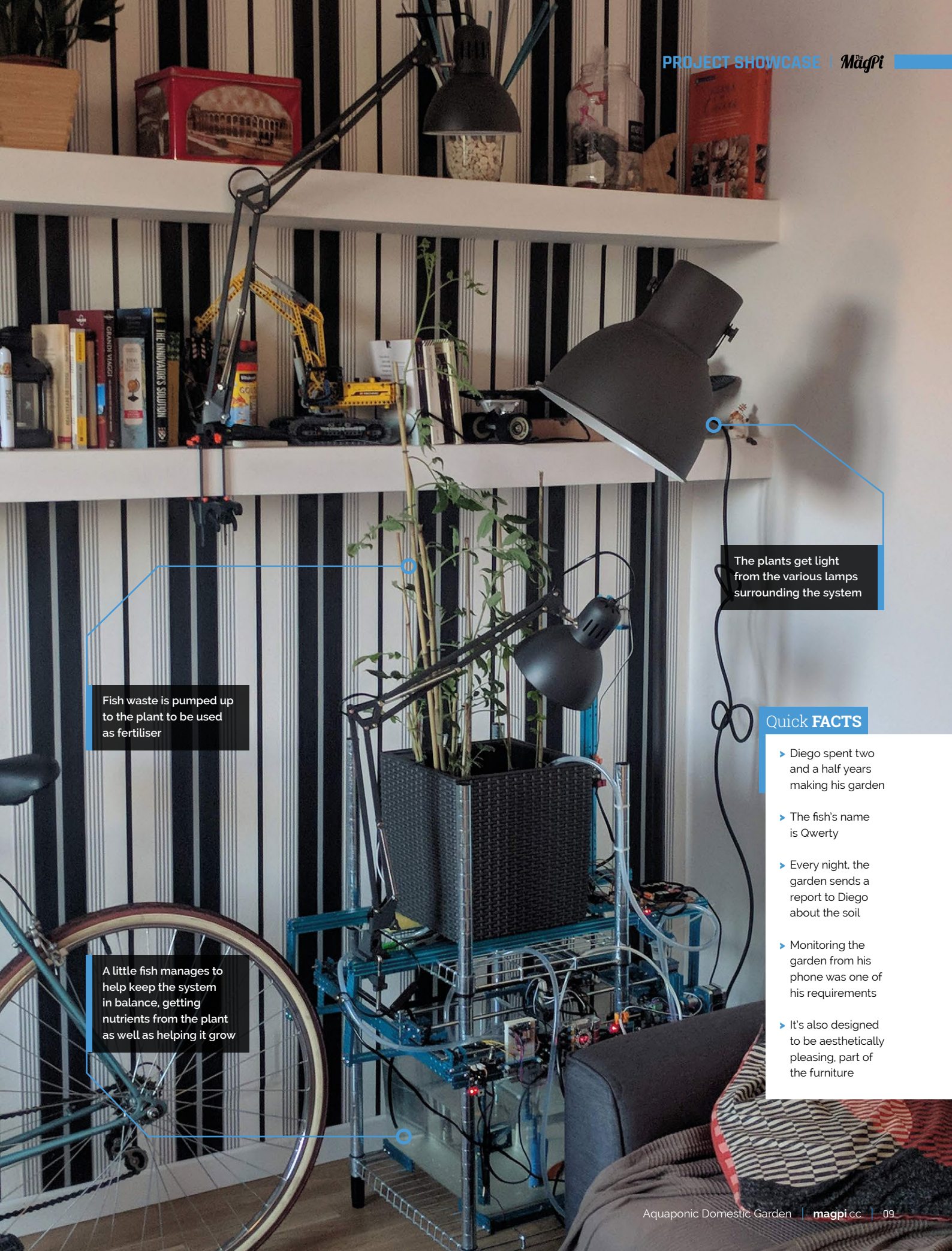
"Living in a big city has pros and cons," Diego Braga tells us. "One of the cons I suffered most is not having a small organic garden."

Diego lives in Milan, in a small apartment with a smaller balcony that he spends a lot of time away from. That didn't stop him wanting to grow some fresh veg for himself, though, and he looked at hydroponic gardens to see if that was a solution.

"One of the cons of this approach is that you need to feed fertiliser in the water, and that's something that, even if it can be automated, is not self-sufficient," Diego explains. "In 2015, EXPO Italy took place and in a pavilion there was an aquaponics system in which plants and fishes were living close and feeding each other. I started to study the aquaponics system and it blew my mind: two different ecosystems (the aquarium with fishes and plants in the solid earth) can feed each other. The waste of one is the nourishment of the other. That was what I wanted."

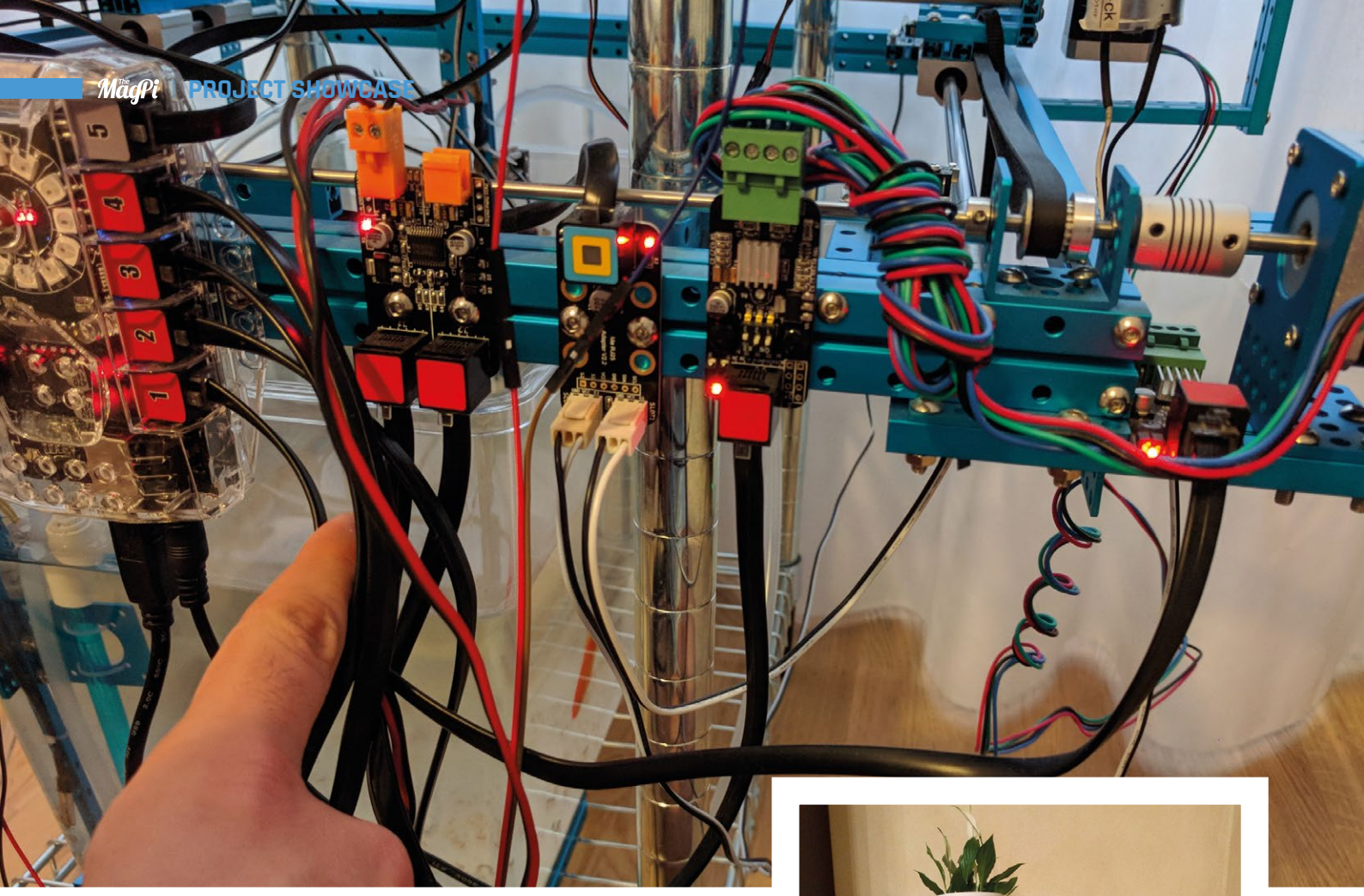


Sitting in a corner of the living room, the mini garden takes up very little space


The plants get light from the various lamps surrounding the system
Fish waste is pumped up to the plant to be used as fertiliser
A little fish manages to help keep the system in balance, getting nutrients from the plant as well as helping it grow

Quick **FACTS**

- Diego spent two and a half years making his garden
- The fish's name is Qwerty
- Every night, the garden sends a report to Diego about the soil
- Monitoring the garden from his phone was one of his requirements
- It's also designed to be aesthetically pleasing, part of the furniture

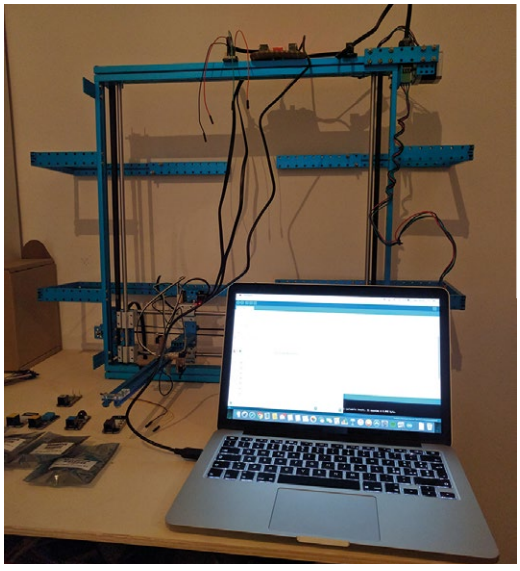


There are a lot of sensors required to keep the system working well

A prototype setup for the system using IKEA parts

Diego figured all he'd then have to automate is the cleanliness of the aquarium, and decided to do it and collect sensor data via IBM Watson's IoT platform. It took him a while to figure out how.

"First of all, I had to study a solution to keep the aquarium clean. The fish waste is the perfect nourishment for the plants and this waste lies on the bottom of the aquarium. One night I had an inspiration: use an X-Y plotter and, instead of writing on a surface, use a submersible pump to collect all the waste and deliver it to the plants."



Building the shelves to house everything properly takes some custom work

Pump it up

Using an IKEA shelf with the aquarium at the bottom and pot on the shelf above, the waste water is pumped up to the plant, which then uses it and pH-balances the water, while gravity allows nutrients to drop down to the fish.

▲ The completed garden in its new home

To get data from the sensors so he could keep an eye on it, Diego used a Raspberry Pi: "I already worked with Raspberry Pi devices in previous projects, but in this case the availability of the Node-RED orchestration that can interact with

> "It works really well. It's wonderful and reassuring to see plants growing in the living room next to the sofa"

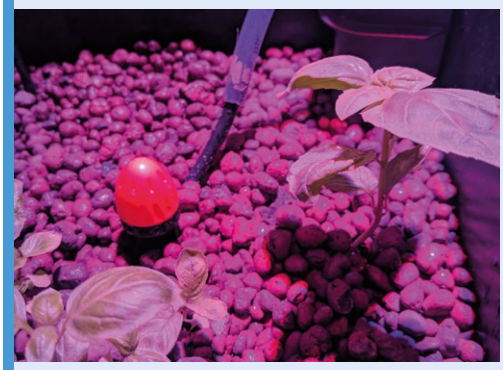
the GPIO pins and the availability of a Watson IoT node library was perfect to connect my aquaponics system to the cloud. The Raspberry Pi was the perfect piece to complete the puzzle."

After so long, what does Diego think of his creation? "It works really well. It's wonderful and reassuring to see plants growing in the living room next to the sofa. Up to now, I've planted basil and tomatoes and they've been really tasty. When I'm sitting in the living room, I can also smell the vegetables."

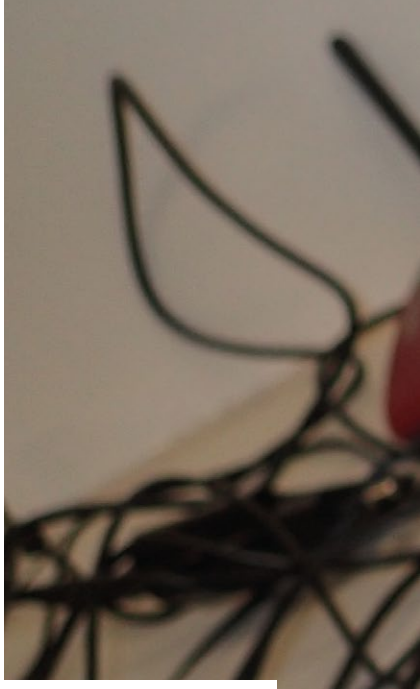
Sowing seeds

01 Let the fish become comfortable with its new home for three to four weeks. The pot doesn't have seeds, but the water is enriching the soil.

02 Based on the chosen seeds, plant them according to the recommended depth and distance from each other. The aquaponic system is inside the apartment so there won't be any weather issues, but Diego prefers to plant seasonally anyway.



03 Make sure plenty of light is being provided to the seeds. You can get special plant lamps, like Diego has used, which help the seeds grow.



PiCam **Marine**

Scientist Dr Autun Purser needed a small device to take photos of cold-water corals on the seafloor, so he turned to the Pi Zero W, as **David Crookes** discovers

MAKER

Dr Autun Purser

Dr Autun Purser researched corals for his PhD. He now tests and develops equipment and methodologies at the Alfred Wegener Institute for Polar and Marine Research.

awi.de/en

Photos courtesy of GEOMAR JAGO team

Ecologists in Germany are deploying camera-equipped Pi Zero Ws off the coast of Norway to discover more about coral activity. Dr Autun Purser works in the Deep Sea Ecology and Technology group of the Alfred Wegener Institute. The group has a keen interest in cold-water corals which are found in most European seas.

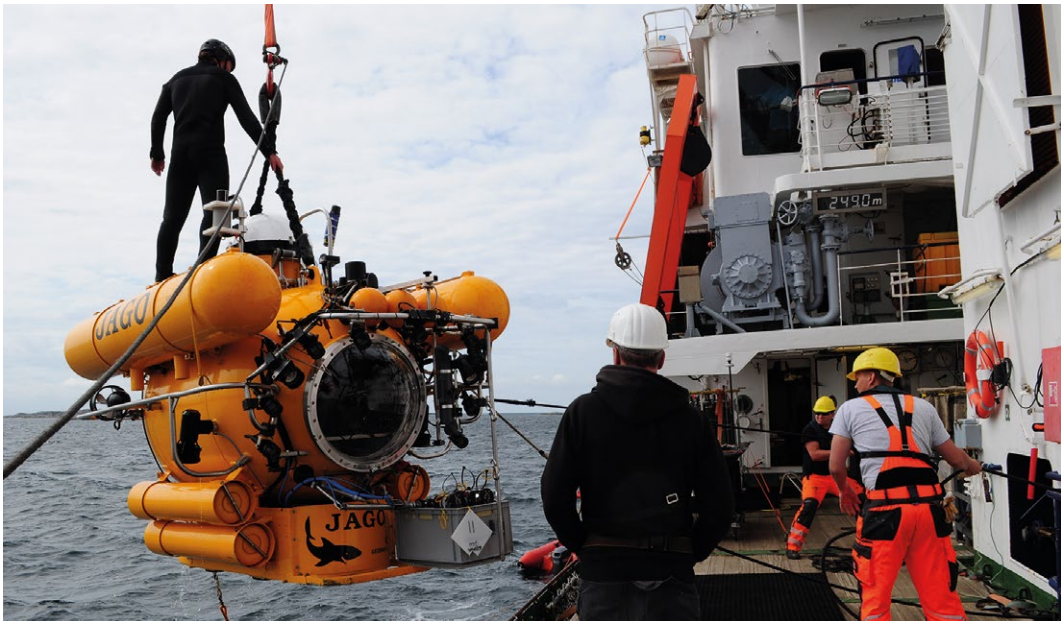
"In the last three decades, we've started to understand these can form reefs whenever conditions are suitable for growth," explains Autun. "During our cruise in the Skagerrak, we intended to map corals and see when, and under what conditions, they did most feeding."

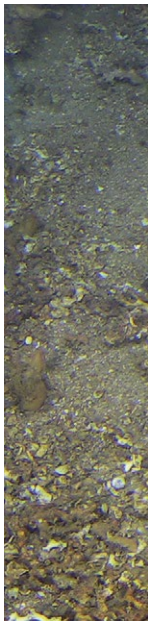
Feeding time

Their aim was to continue the development of "cheap camera systems which can be used for a range of applications in the deep sea, down to depths of at least 6000 metres. We investigated the use of Pi Zero W computers and [Pi Camera Modules] to record video snippets of both the sea-floor and any scientific devices that we place underwater, and we found the small size of the computers to be of great benefit to us."

The Pi Zero Ws and cameras are placed in strong, waterproof pressure containers, and powered by Li-ion batteries that can withstand the cold deep ocean conditions. "The WiFi connectivity allowed us to set up a router on deck to both initiate our cameras and, on retrieval from the sea-floor, download our collected images without having to reopen the pressure housings," reveals Autun.

He and two colleagues programmed the camera system using Python 3 to turn on an LED light and take a maximum resolution image, at set times. It has proven "capable of imaging individual corals from 2 m distance, allowing us to tell if the tentacles were actively extended or not."



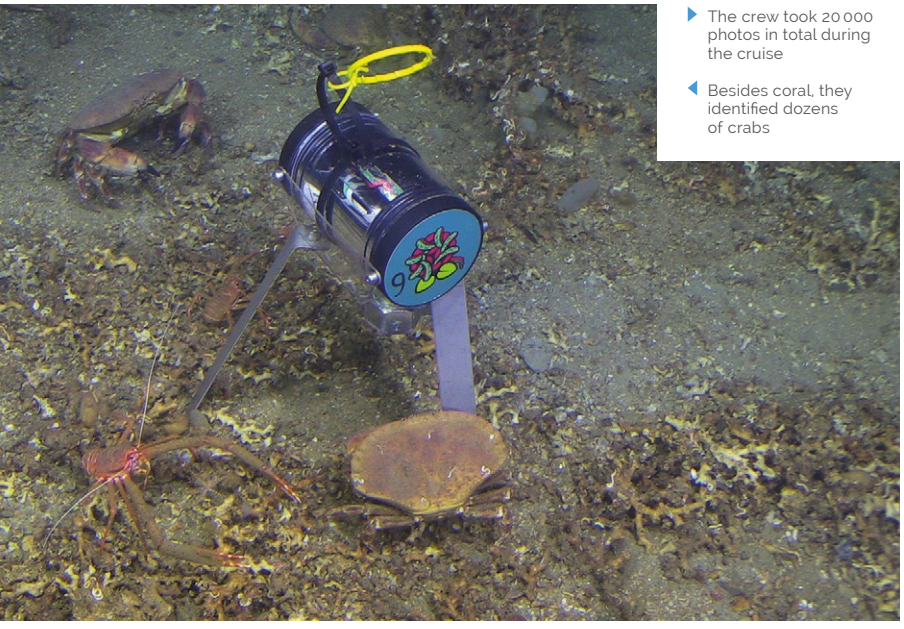


The PiCam Marines are sent underwater in the deployment basket of a submarine. The captain, crew, and scientists aboard RV Poseidon cruise POS526 were also essential for the initial deployments

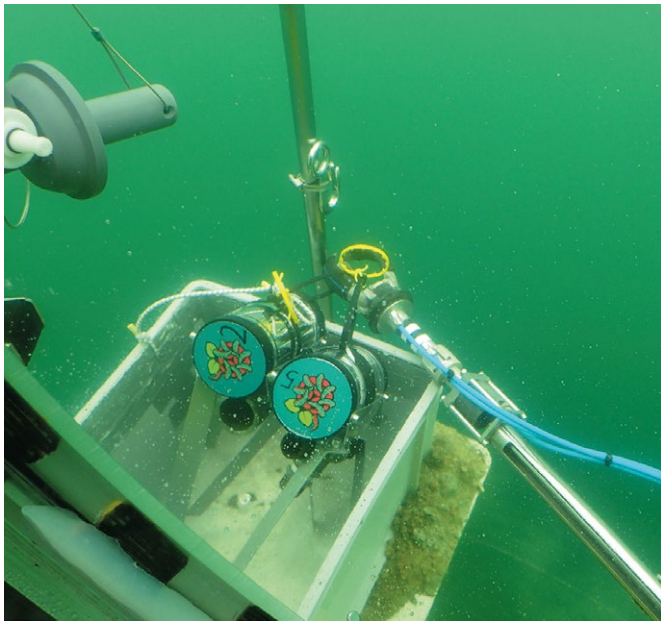
Two lithium-ion batteries have enough capacity to take an image every five minutes during a ten-day deployment

The components fit within a transparent plastic tube 15 cm in length and 9 cm diameter, with machined top and bottom sections

The Pi Zero W, camera, LED flash, and real-time clock chip (to ensure the system keeps the correct time) are attached to a plastic frame



- The crew took 20 000 photos in total during the cruise
- Besides coral, they identified dozens of crabs



FRILLER

This amazing robot changes the shape of its wheels to overcome obstacles, as **Phil King** discovers

MAKER

Al Bencomo

Al is a researcher in the Intelligent Systems Division at NASA Ames in California, developing and testing algorithms for autonomous unmanned vehicles in the real world.

magpi.cc/UpZKrP

A two-wheeled robot scoots under a glass table, turns around, and... its wheels expand into spiky petal-like 'legs' for a larger radius, enabling it to clamber over the top of the table. This is the phenomenal FRILLER, created by Al Bencomo and based on an original body design by Carter Hurd.

"The idea was inspired by the STAR robot from UC Berkeley and some of the research done at [NASA] JPL," Al reveals, "which includes prototypes of collapsible robots and robots with tails."

While the wheel transformation process looks complex, Al tells us that the 3D-printed design and mechanism is a lot simpler than we'd expect. "The first proof-of-concept trial had a tendon-like connector in order to tension the wedges in one state or the other. However, the latest version relies on interlocking sections, which can be 3D-printed without having to print support structures."

Rough terrain

Solarbotics GM3 224:1 gear motors are used for the wheel transformation, each one able to provide 0.343 Nm of torque at 3 V. Combined with the spiky expanded wheels, 1.2 Nm 12 V advancing motors, and a flat fishtail to aid balance, this enables FRILLER to traverse a range of rugged terrains: so far, it has been tested on snow, dirt, grassland, and small gravel.

"FRILLER's preliminary field tests haven't been particularly challenging," admits Al, "but they can still be counted as a success. Future tests will include areas such as steep slopes and sand dunes."

Al says the deformable wheel concept has huge potentials for serious practical uses in fields like geology, "since it can be used in hard-to-reach locations like volcano ledges, for example. It could also be used as a scout for larger rovers to enhance the way they currently do exploration science."

3D-printed design

The FRILLER project took Al 80 person-hours to build and program. "Normally, it would have been complicated and long-winded to create the resulting robot, but 3D printing saves a lot of time." All the 3D print STL files and code are open-source if you want to build your own: **magpi.cc/pYSbeV**.

The round wheels transform into spiky legs, enabling the robot to tackle rough terrain

At FRILLER's heart is a Raspberry Pi 3 running Android Things. The TouchOSC app on an Android phone is used to manually control the robot. "During field tests, the Android device is set up as an access point (hotspot) and FRILLER connects to it to receive commands via UDP packets."

Al plans to add autonomous control to conduct science beyond line-of-sight. He plans other improvements too: "The next step is adding some sensors like a spectrometer to study the chemical makeup of its environment, and GPS to help navigation when you're not directly guiding it."

Quick FACTS

- It took 80 person-hours to build and code
- Build details are at **magpi.cc/UpZKrP**
- While AI works for NASA, this is not an official project
- It has a 3.5Ah 11.1V Li-ion battery
- Range is ~650m on a level dirt path

A Pi 3 equipped with an Adafruit Motor HAT is the brain of the robot

FRILLER's flat fishtail aids stability while it's climbing

▼ The robot's wheels remain round (left) for faster travel on roads or indoors, but transform into spikes (right) to overcome obstacles off-road

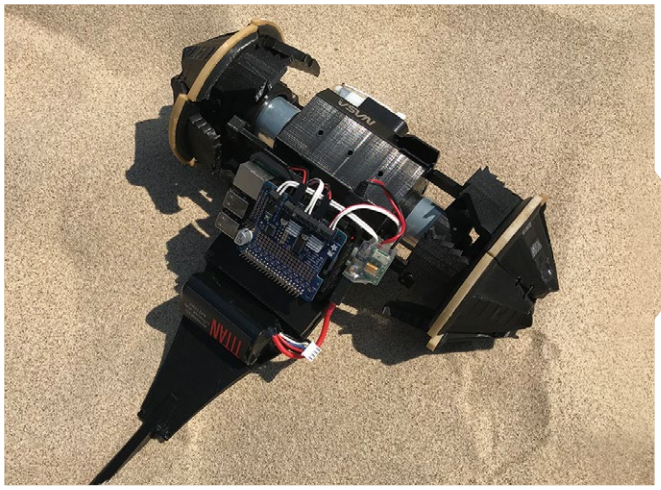


Table Foosball

If football is a results business, it should be easy to keep score. With Matmi's Pi-driven scoreboard for table football, you can do just that. 'Result,' reckons **David Crookes**

MAKER

Jeff Coghlan

Jeff Coghlan founded Matmi in 2001 and he has worked with some of tech's biggest legends including Steve Wozniak and Ian Livingstone. He is a Manchester United fan.

matmi.com

Football is a funny old game, so they say, but table football – or foosball as Americans call it – is much quirkier. Popular for many decades (its origins stretch back to 1921), it involves moving up to eight rows of footballers on rotating and sliding bars in the hope of passing the ball from player to player and smashing home more goals than the opponent.

Microswitches are fitted within the foosball table pipes and they are connected to the Pi's GPIO pins

> We decided to reimagine our foosball table as a technologically enhanced experience

The bods at Matmi understand it well. When they're not busy working on digital projects for the likes of indie band Gorillaz, singer Lily Allen, actor Warwick Davies, the BBC, Alton Towers, United Airlines, Comic Relief, and more, they gather around their own foosball table in the office and have some fun.

Trouble is, they get so involved, they often forget the score, and those who are watching struggle to keep up with the action. To solve this, they turned to the Raspberry Pi 3B+. "We decided to reimagine our foosball table as a technologically enhanced experience, creating competition and adding scoreboards," says Matmi founder Jeff Coghlan.

The idea was to create a Red vs Blue Table Foosball companion app which lets players enter their names and track their scores in real-time. Initially, the team worked on sketches and analysed the table to work out the best way to detect goals. They quickly identified most of the required hardware and ordered a Pi, a Class 10 microSD card, a USB microSD interface, a 2.5A power supply, some microswitches, and lots of wires.

"The Raspberry Pi 3B+ fit the requirements perfectly," Jeff says. "Not only is it a GPIO-capable single-board computer, but when using a Raspbian desktop and Visual Studio Code, it is also a fully capable development suite, making the development process that much faster and more convenient."

Goal-line technology

The physical side of the project was also completed quickly, with sensors placed within the foosball table's pipes to update the scores whenever the ball rolls past. "We examined various options for goal sensing and ultimately decided to use microswitches with paddles to detect the goal as the ball drops down," Jeff explains. "Gravity forces the paddle out of the way, triggering the microswitch and this simple approach works perfectly. It allowed us to quickly move on from the physical and get stuck into the presentation."

The display runs from a laptop. Goals are celebrated with a sound clip and on-screen confetti, and the scoreline is updated

When a goal is scored, the signals are read by a program running in Visual Studio Code installed on the Pi

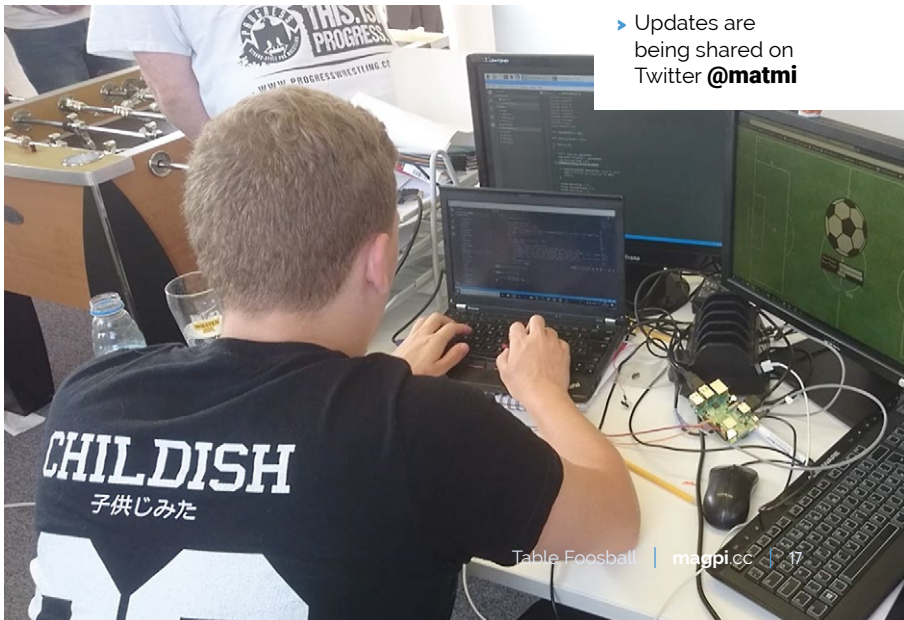
Quick **FACTS**

- The project should work with any foosball table
- It details who scored and when
- The display runs from a WiFi-connected laptop
- Matmi wants to add a tournament mode
- Updates are being shared on Twitter **@matmi**

▼ The team says using the Vue.js open-source JavaScript framework made the process of creating the app easier due to the abundance of examples on Stack Overflow

Will Booth, who started in the Matmi internship program, began by setting up the Pi to read the sensor inputs and print them to a score log. A laptop was then connected to the Pi via a WiFi network, allowing the data to be presented on the Pi's HTTP server. "The built-in wireless adapter and Raspbian OS made configuring the network easy and it also fit in with our end vision of preserving the vintage feel through minimal wiring," says Will, who also designed the scoreboard's graphics using Paint.net.

The result is a table which lets the players and spectators follow the action, but the team is keen to go further. "For now we are working on a top-down camera for goal replays and an evolution of the GUI," says Jeff. "We will present it again when we have taken it to the next level."



Rock, Paper, Scissors

Rock, paper, scissors for the modern day? **Nicola King** investigates a very contemporary twist on an old favourite

MAKER

Julien de la Bruère-Terreault

A Canadian maker and engineer, whose interest in computers began in the early 1980s when he received a Texas Instruments TI-99 computer for his birthday.

magpi.cc/zAzFgs

The traditional game of rock, paper, scissors has been given a very modern slant by Canadian aerospace engineer Julien de la Bruère-Terreault. He has developed a computer-vision and machine-learning-driven version, pitting one player against a computer.

"The idea for the project actually comes from my son," reveals Julien. After coding a text-based rock, paper, scissors game in Python, Julien was experimenting with basic computer vision on his Raspberry Pi when his son asked if he could make a version of the game that uses the camera to detect the hand gestures. "Knowing it was feasible, I accepted the challenge, seeing this as an opportunity to learn a lot in the process."

Over the course of a year, Julien developed his skills around machine learning and computer vision. "The challenge for me in this project is that I had to learn almost everything required to make it work," he says. "Developing and testing the machine-learning algorithm was the most difficult part." This algorithm, or 'classifier', teaches the computer to recognise a set of hand gestures: "[It] has been previously 'trained' on a bank of labelled images corresponding to the rock, paper, and scissors hand gestures."

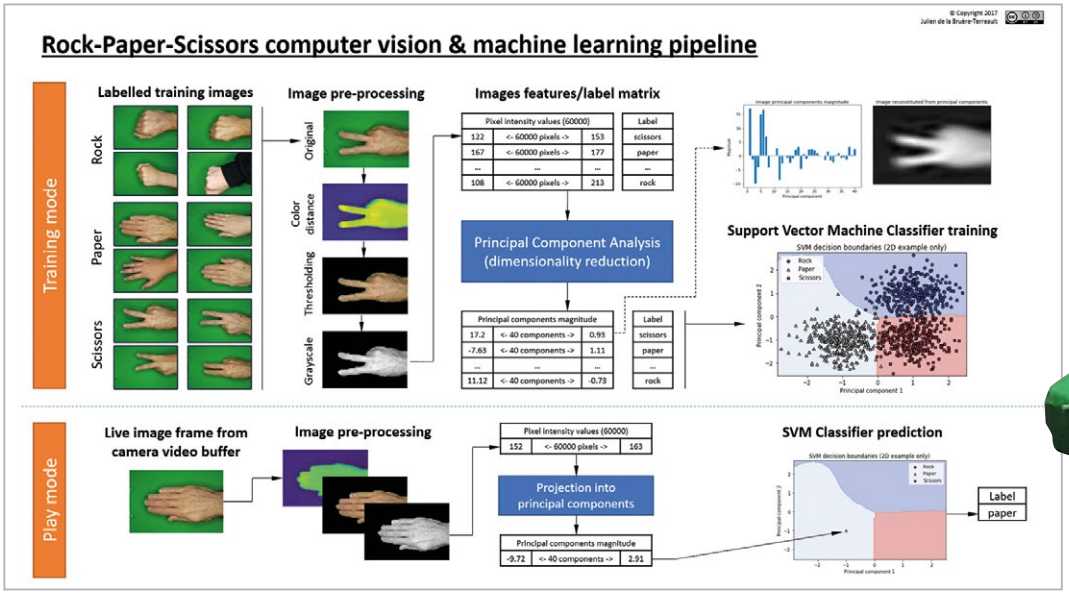
Training images

Julien began the project by designing the 3D-printed fixture (**magpi.cc/VqvUAJ**) to hold the camera and LED lighting strips. He then developed a simple application to capture training images to develop the machine-learning algorithm.

"I started with an initial set of approximately 150 labelled images." As this was quite a low number, "I had to select a relatively simple algorithm with few parameters to tune and perform dimensionality reduction on the images."

Once happy with the algorithm, Julien began work on enabling the game to be played in real-time, adding the logic and graphical interface.

He says the Raspberry Pi was an integral part of the build: "With its Camera Module and great Picamera library, its portable size and sufficient computing power, the Pi was the ideal platform to develop this project."

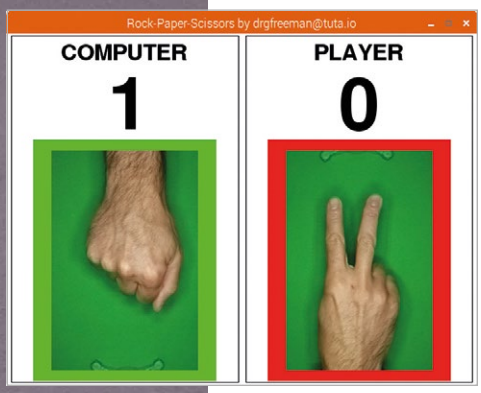


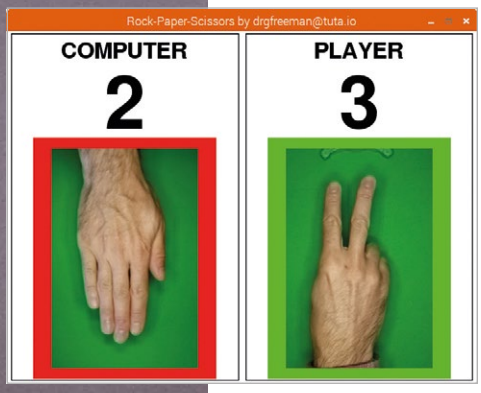
The processing pipeline for the training of the image classifier (top part) and the prediction of gesture for images captured during the game (bottom part)

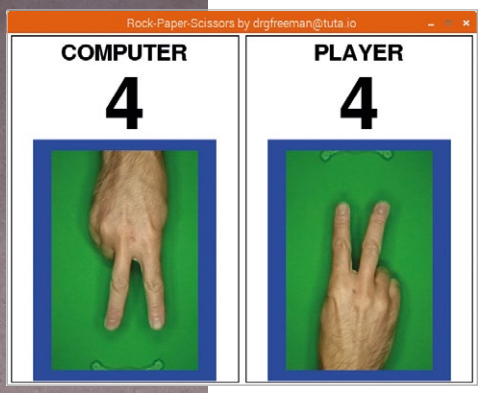
Held by a 3D-printed support, the camera is surrounded by LED strips for consistent lighting

A Raspberry Pi 3 uses OpenCV to interpret hand gestures

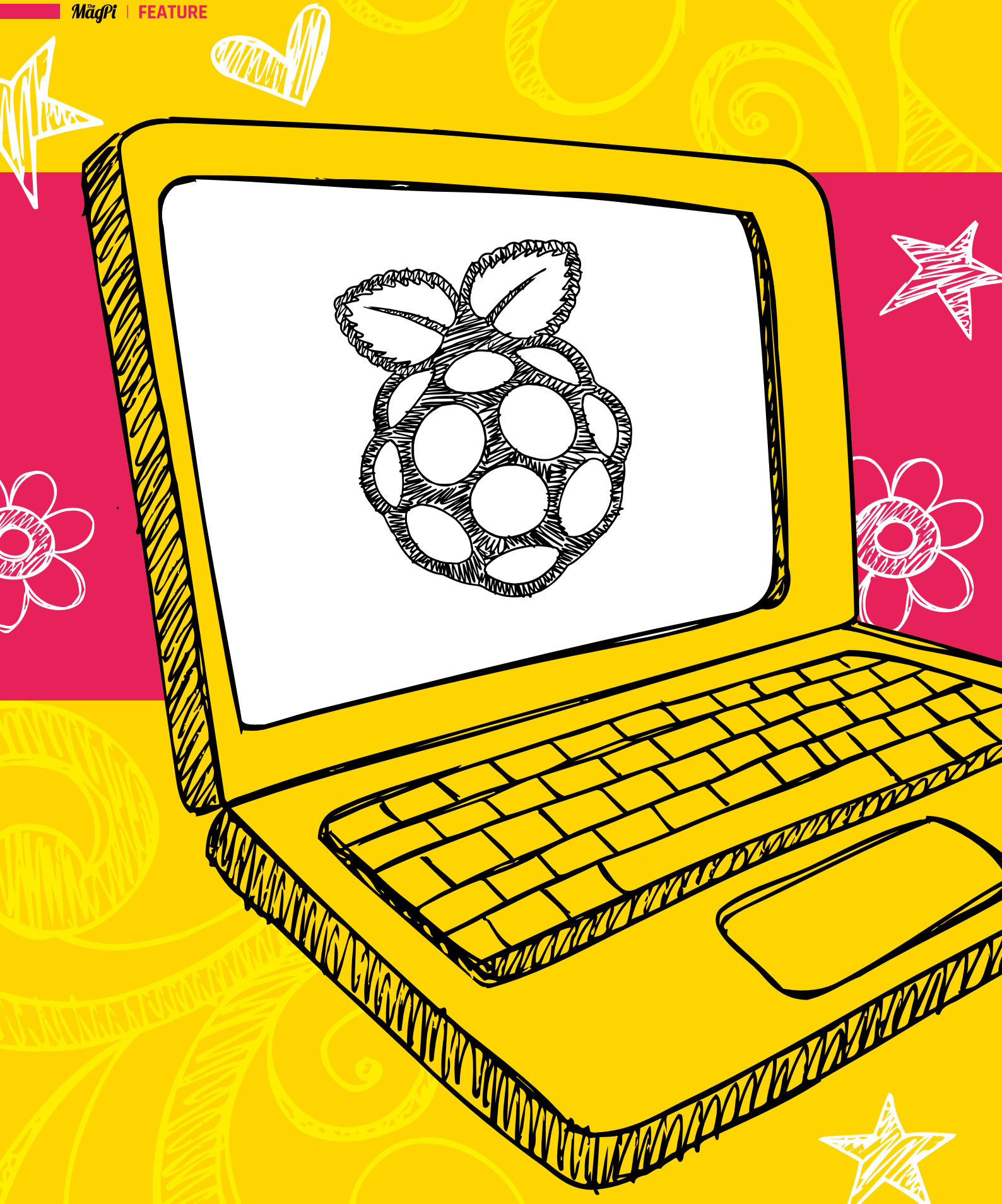
The green base enables background subtraction in the captured images





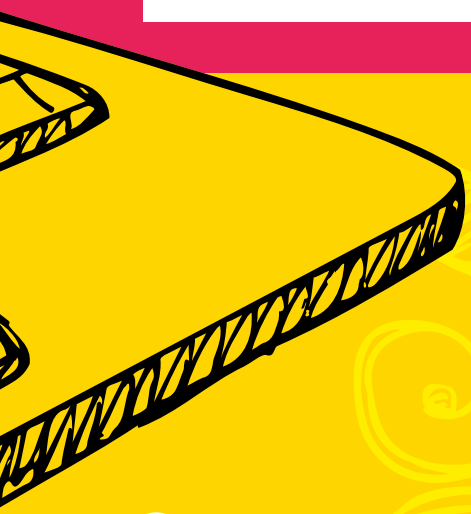


▲ During the game, the player makes a gesture under the camera, and the Pi selects a random one of its own



BUILD A LAPTOP

Digital making on the go with Raspberry Pi



The Raspberry Pi's size and relative power enables it to be both a useful desktop computer, and one that you can slip in your pocket (if you really wanted to).

Picking up your Pi when you head out of the door is one thing; hooking it up to a monitor and keyboard and such is quite another. What if you could pick up your Pi and this was all attached? That would be a Pi laptop.

There are many ways you can create a Raspberry Pi laptop, such as custom builds, hardware recycling, and even just buying a laptop based on a Pi. If you're interested in the idea but not sure which one to choose, you'll be able to learn about the ins-and-outs of all these options over the next few pages. Let's get portable.

MAKE YOUR OWN RASPBERRY PI LAPTOP

Just a few components are needed to create a take-anywhere Raspberry Pi

You'll Need

- PiTFT 3.5 **adafru.it/eno**
- Riii miniature keyboard **magpi.cc/DVuEib**
- PowerBoost 1000C **adafru.it/2465**
- 2000 mAh 3.7V LiPo battery **magpi.cc/tLqhTH**
- Mini metal speaker **adafru.it/1890**
- PAM8302 amplifier **adafru.it/2130**
- 3D printer (or use a service)
- SPDT switch
- Screws (4-40 and 2-56) and wires

One of the first things people often remark on when seeing a Raspberry Pi is its size, and that goes double for the Pi Zero. Yet, despite their diminutive form factor, Pi boards can be surprisingly complex to convert into a usable laptop, mainly due to their rugged construction and connectors. Here, we'll overcome some of the obstacles with a bit of clever design and create a useful, portable handheld computer.

01 Print the case

This build is based on an Adafruit project, which provides 3D print files for the enclosure that are downloadable from magpi.cc/fYoDzm. Thingiverse can print them for you, but if you have a 3D printer to hand, you're set. Expect about an eight-hour print time for all the parts.

Don't worry about assembling the case just yet, but test everything out. Make sure the hinges fit 'Lego-style' to the case and slot together comfortably.

02 Prepare the PiTFT

The PiTFT features a breakout of many of the Raspberry Pi's GPIO pins. We can use this to provide power to the Pi. Bend all the breakout pins on the edge of the PiTFT as flat as you can, or snip them off, but not right to the base. Either connect jumpers, or solder wires, to pins 2 (5V) and 6 (GND). Leave plenty of length.

03 Mount the battery

Wrap the battery in some duct tape to keep it safe, then lay it along the Pi, wires away from the USB and Ethernet ports. Now attach the PiTFT to the GPIO header of the Pi. You should end up with a battery 'sandwich' with the battery's wires free of the assembly. Be careful that the battery is safely insulated from the Pi's components.

The case design is based on the Toshiba Libretto range popular in the 1990s

Access to USB and Ethernet

Wireless keyboard means no wires!

04 Thread the wires and mount the speaker

Before you insert the assembly into the printed case, run two wires (one for positive, one negative) between the Pi and the PiTFT. Don't worry about length: too much is a good idea at this stage. The speaker may need its wires extending to reach the amplifier at the other end of the casing. Once you're happy, snap the speaker (carefully!) into the mount on the case and screw the PiTFT and Raspberry Pi to the four pillars around the screen aperture.

05 Hook up the sound

Using the pair of the wires you threaded through the Pi 'sandwich', solder them to the 'Vin' (positive) and 'Gnd' (negative) points on the amplifier. At the other end, following the circuit diagram (**Figure 1**), solder the same pair to the ground and 5V lines on the PowerBoost, shortening as required.

Next, connect the speaker's wires to the amplifier's output. You may need to extend them to fit. Finally, and very carefully, solder two wires from the A+ and A– lines on the amplifier to the Raspberry Pi board, as shown in **Figure 1**.

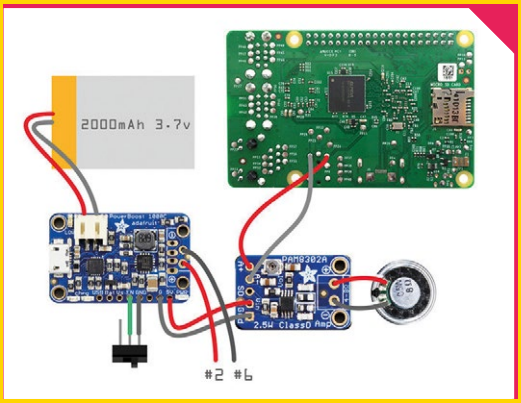
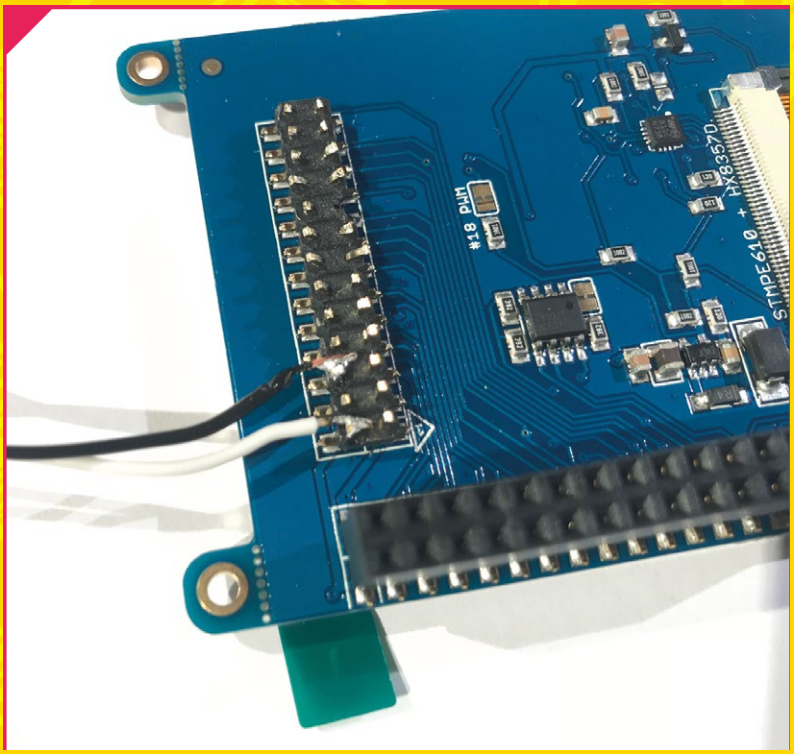


Image credit: Adafruit

Figure 1 There's no complicated circuitry to assemble here, just power lines between the various components



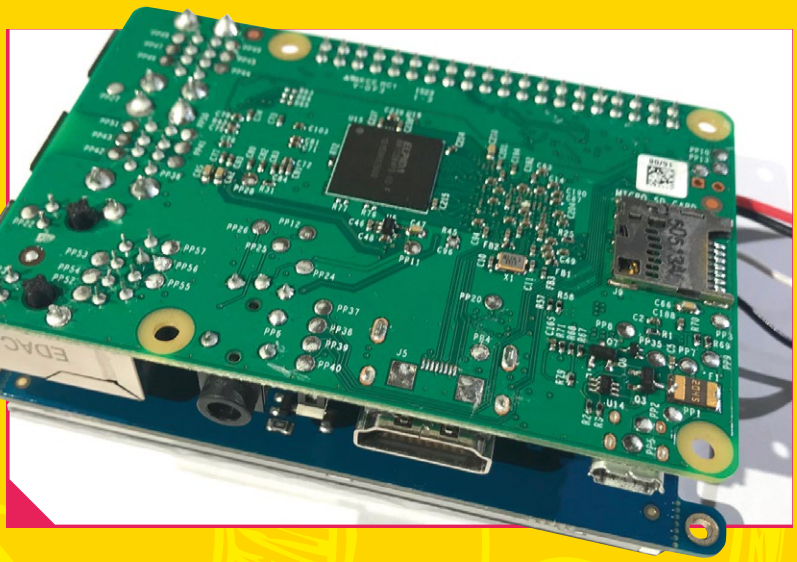
▲ Power to the Raspberry Pi is provided though the PiTFT screen. Either bend the pins back or snip them. Then solder wires to 5V and GND

06 Power

The PowerBoost will deliver power to the whole system from the battery and also manage recharging. We're powering the Pi through the PiTFT, so using the wires we soldered on to it earlier, connect them to the PowerBoost as shown in **Figure 1**, having shortened the wires to fit.

Solder two further wires to EN and GND on the PowerBoost. These need to connect to the SPDT switch. Thread the wires through the opening for the switch, solder one to the centre pin and another to one of the sides.

▼ Connect the PiTFT and Raspberry Pi together with the battery in between



07 Software

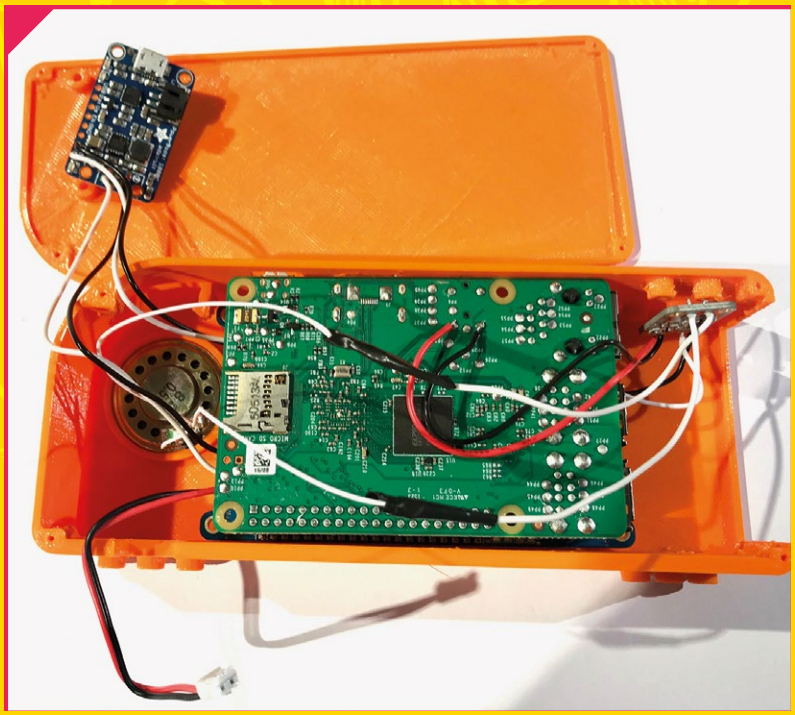
Before going any further, get your operating system ready to go. Raspbian can't talk to the PiTFT by default, so connect a screen, keyboard and mouse, prep your microSD card with the latest version of Raspbian, and use the Terminal commands listed at **magpi.cc/XaeUem** to install the necessary software to drive the screen. You need to set the screen to rotate 270° and to mirror the HDMI output. Once rebooted, your PiTFT will come to life.

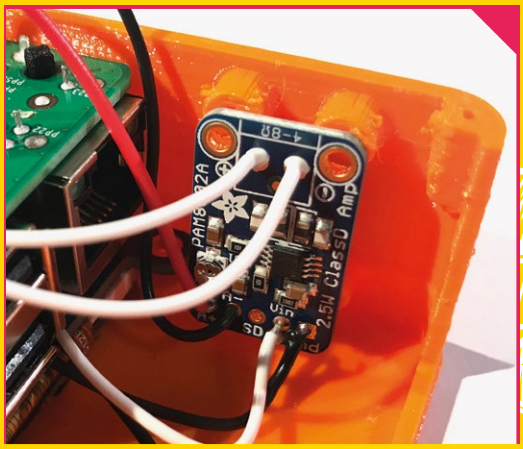
08 Main body assembly

Carefully secure the amplifier with 4-40 machine screws. Lay the case back alongside the main body and screw in the PowerBoost. Attach the battery (and then switch off if everything comes to life). Insert a USB WiFi adapter (if required) and the USB receiver for the keyboard. Tuck all the wires away carefully, looking out for shorts. Push the switch into the clips provided. Now place the back cover over the main body and secure with 2-56 machine screws.

WANT A POCKET COMPUTER?

There are many other options for building a Raspberry Pi laptop. Sites such as Thingiverse have several plans for larger assemblies, many using the official 9-inch touchscreen. Capabilities vary, so make sure you have a look around and select one that's perfect for your needs. If you want to go even smaller than this project, the Nano Pi2 (**magpi.cc/TUULny**) is about as miniature as you can go and fits snugly in the palm of your hand.





▲ The Pi, battery, and screen are in place. Make sure the remaining components are connected as the circuit diagram (**Figure 1**) shows, then secure them to the case

09 Final assembly

It's time to finish everything off. Snap the keyboard into the printed mount. Having checked everything fits and double-checked orientation, glue the hinges to the body and keyboard mount, then allow to dry. Push the hinges together, then secure each with long machine screws.

Check all the ports line up. Turn on the keyboard, flick the power switch, and after a few seconds the screen should come to life. Congratulations, you have a handheld, rechargeable Raspberry Pi.

10 Using your laptop Pi

Although the screen is great for gaming, we wouldn't recommend writing your next great novel on it. That said, the density of the screen makes it comfortable for a bit of web browsing or using the Terminal. The bonus of audio makes for a cool little radio too.

You can power the system using the usual connector on the Pi, but this will not charge the battery. Instead, connect to the micro USB port at the base on the main unit. Don't forget, you'll need to keep the keyboard topped up too.

If you find the Raspberry Pi does not appear to be shutting down correctly, don't worry. The screen does not refresh correctly on shutdown so appears to freeze. Just wait a few seconds, then switch off.

READY-TO-GO ALTERNATIVES

If building your own Raspberry Pi laptop doesn't float your boat, but the idea of a carry-anywhere little laptop appeals, there are two key players in the market who will provide you with an (almost) complete laptop. Pi-top is an established player and the second version of its laptop is smart, elegant, and practical, with 6–8 hours' battery life.

Whilst not strictly a laptop, the Kano Computer Kit is extremely portable and easy for small hands to build. Everything you need is provided and assembles into the screen case, which includes a battery, leaving only the keyboard (with integrated trackpad).

KANO

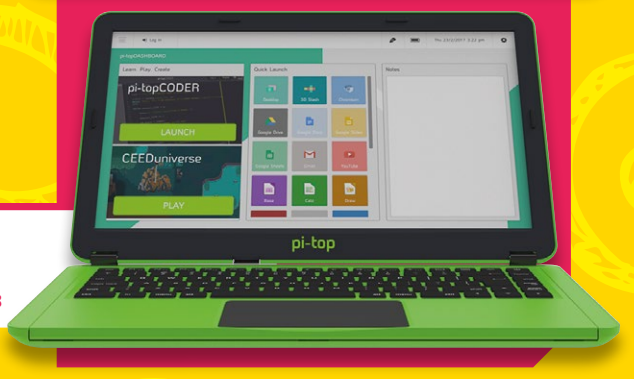
▶ **kano.me**

Priced from: **£213**

PI-TOP

▶ **pi-top.com**

Priced from: **£243**



RECYCLING OLD LAPTOPS

Some assembly is required, but it can be done

Got a doohickey from the Stone Age lying around? Loath to see it go to landfill? Maybe you've wondered if you can revive it with a Raspberry Pi? After all, there's a nice keyboard, big hard disk, and lovely screen. Surely we can just plug those in and off we go? Well, no.

The chief obstacle is design. Laptops need to be as compact as possible to appeal to the market and one of the easiest ways to do that is by creating proprietary systems, setting the standards and sizes you need. A classic example is the screen interface. Standards used by laptops vary by manufacturer: some use LVDS, some not. The same goes for the software driving them and the actual connectors. Additionally, a lot of drivers are closed-source, so Raspberry Pi boards lack the necessary software to control various parts of a potential donor laptop.

> "As ever in the Pi community, the more, erm, tenacious amongst us have risen to the challenge"

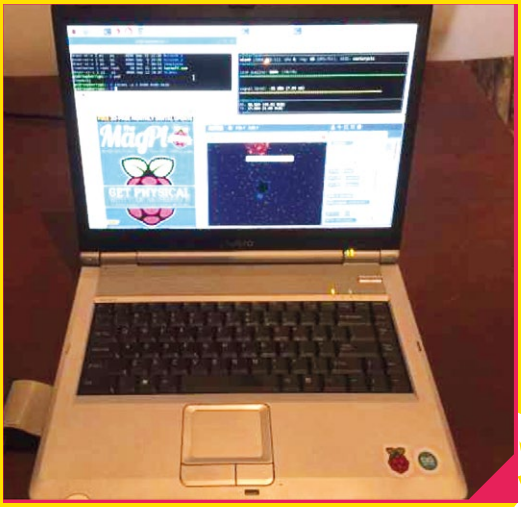
It's not all bad news, though. As ever in the Pi community, the more, erm, tenacious amongst us have risen to the challenge. The results vary, with some very clever solutions alongside some more out-of-the-box thinking. Let's take a look at some of our favourites.

THE PI-TEENSY LAPTOP

▶ **magpi.cc/PsLXkR**

Frank Adams wasn't taking no for an answer. His solution was to look back to earlier generations of laptops, a time that was significantly clunkier that today's razor-sharp edges. Clunky means space and that means a Raspberry Pi can get inside the case. A very early Sony Vaio was the perfect candidate.

This complex build addressed the typical problems of interfacing with laptops in creative ways. The keyboard was wired up to a Teensy microcontroller which, with a little code to scan the keyboard's wiring matrix, converted it into a USB keyboard. The Vaio featured a standard LVDS interface, so an off-the-shelf video controller board turned that into HDMI. Not stopping there, Frank made sure the various components such as LEDs, WiFi antennas, and fans were repurposed too. He even eventually got the Vaio's own battery talking to the Pi and supplying power. Even the GPIO is still usable.



MOTOROLA ATRIX DOCK

▶ **magpi.cc/hnWChv**

Here's a classic example of practical upcycling. Released in 2011, Motorola's 4G Atrix Android phone was a contender to Apple's dominance. Its differentiator was the Atrix Dock, a screen, keyboard, and battery assembly that connected to the Atrix phone to convert it into a netbook-sized computer. It was a miserable failure due to cost. Not only was the dock priced at around $500, but you then had to buy the phone as well for it to be of any use. With decent netbooks selling at around $300, there really was no business model for it.

> "The Atrix Dock can now be found significantly cheaper and the connectors on it are standard"

The good news? The Atrix Dock can now be found significantly cheaper and the connectors on it are standard: a Micro HDMI connector for the screen and a micro USB connector for keyboard and power. Almost perfect. All you need to do is a little bit of cable work to connect everything up and you have a great solution that includes battery power. The Atrix Dock itself is very nicely designed and with a bit of creative thinking, you can mount the Pi behind it or even just leave it freestanding.

Dedicated tinkerer 'thegrendel' has published a comprehensive guide to the types of cables needed. This is a great alternative if physical hacking doesn't appeal and, of course, you can find an Atrix Dock for sale.

THE VENTI-PI

▶ **magpi.cc/bnoiTy**

Here's a true left-field idea. If you've got an old netbook to hand, all you need to do is get a copy of Linux running on it with a VNC client. Then use that to connect to a Raspberry Pi running in… a coffee cup? Dave Chew took a Starbucks Venti-sized reusable cup and installed not only a Raspberry Pi but also a one terabyte drive. He's even provided 3D-printable parts to create a stable structure for everything to live in. The cup sits innocently next to the netbook, its only give-away being the USB power cable.

This isn't a new idea (although the execution certainly is). Rather than getting all messed up in tiny electronics, netbooks such as the Asus Eee series popular in the late-2000s can now find use as an update to the classic 'dumb terminal', providing a screen and keyboard for your Raspberry Pi in a neat package. Pi Zeros, with their lower power requirement, can even be powered by them.

RASPBERRY PI ON X86 LAPTOPS

Raspberry is not the only fruit...

If what you want, what you really, really want, is the Raspberry Pi desktop experience but on a laptop, there's another route to consider. In late 2016, the Raspberry Pi Foundation launched a build of Raspbian, including the full desktop environment, for x86 processors. That (probably) means your laptop. When it comes to older laptops that may be struggling with what Apple and Microsoft can offer, you might want to look at installing the Raspberry Pi Desktop OS, with no Raspberry Pi required at all. No 3D printing, no soldering, just boot and go.

You would think the biggest disadvantage of this approach is the lack of GPIO, but all is not lost. The exceptionally clever Raspberry Pi Zero, with its OTG (On-The-Go) interface, can act as a USB 'gadget' and this includes remote access to the GPIO pins. So, provided you have a Raspberry Pi Zero to use, you can develop on a laptop and still read and write to the GPIO. This will even work in a virtual machine on your main computer.

If you've been using a Raspberry Pi to learn coding or maybe make games with Scratch, this could be the option to go for if you have an older laptop. Raspberry Pi Desktop OS is extremely lightweight compared to the mainstream operating systems and you may be surprised how well your 'older' laptop performs.

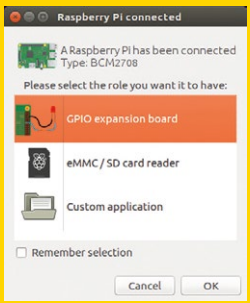
Burn and boot

To get started, you'll need a copy of the latest Raspberry Pi Desktop OS for x86, from **magpi.cc/2j0TqNm**. You'll get an ISO image which can be written to a microSD card or CD-ROM. Which you choose will come down to the target laptop. Older machines don't know how to boot off microSD cards and most netbooks don't have a CD-ROM

drive. Some experimentation may be required. As a rule-of-thumb, pressing **F10** when booting your laptop will display a menu where you can select what you want to do.

If you've ever installed a Linux system before, particularly Debian, the process will look familiar. You even have the option of a 'live CD', booting off the disc or microSD card without installing anything internally. This is great for testing things out. If using a microSD card, you can even boot into 'persistence' mode, where configuration changes and files are written back to the card, so you lose nothing on reboot. If you want to go ahead and install to your internal storage, you will be guided through the full process.

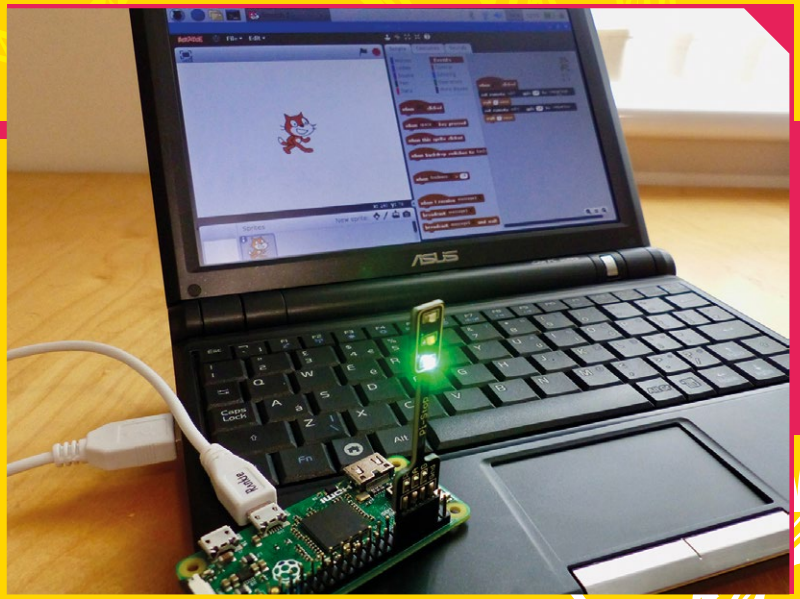
Once installed, the familiar Raspberry Pi desktop will be looking back at you. This worked very well on our test Asus Eee 900, a classic netbook from 2012. Internal components including sound, WiFi, and USB all worked, with no additional configuration required. Your favourite software will run just as if it were on a Raspberry Pi to begin with.



No GPIO? Yes GPIO!

Should you want to get some GPIO action on, it's very simple to do so. The latest release of Raspberry Pi Desktop OS for x86 has everything you need ready to go. You'll need a Raspberry Pi Zero (or Zero W) with no microSD card inserted. Connect the Zero's peripheral micro USB port (labelled 'USB') to a spare USB port on the laptop. The Zero will realise it's now a USB 'gadget', and after a few seconds the operating system will ask you what to do.

Select 'GPIO expansion board' and the Raspberry Pi Zero will be fed some code and rebooted. It uses its OTG interface to become a USB network device on your machine. To talk to it using Python, the GPIO Zero library can access the pins with a little configuration first (see **magpi.cc/2kvGsY5** for instructions). If Scratch is your thing, you're in luck. Fire up Scratch 2, click on 'More Blocks', 'Add an Extension', and select 'Pi GPIO'. The new blocks make it easy to read and control the GPIO pins. You've got all the features of a Raspberry Pi on your laptop!



So is it worth doing this? Well, if you're doing a lot of Raspberry Pi work, this can certainly be a nice way of upcycling old hardware and being able to work with a familiar environment. There are also fewer wires to get tangled up in and a Pi Zero is all you need to add some GPIO goodness.

> Most importantly, Raspberry Pi Desktop OS isn't some 'toy' or inferior operating system: it's built on Debian

Most importantly, Raspberry Pi Desktop OS isn't some 'toy' or inferior operating system: it's built on Debian, making it one of the best supported operating systems out there. The Raspberry Pi Foundation has done extensive work to make this OS friendly yet powerful. The huge range of software (nearly all free) and online support make it a solid choice for anyone, not just Raspberry Pi enthusiasts.

PI ZERO: GADGET MODE

Want a quick way to use a Pi on your laptop? You can connect a Pi Zero to your laptop via a USB port and use VNC to then dial into Raspbian running on the Pi. You need to use a special USB gadget mode for this, but it lets you use the Pi Zero as usual from your laptop. Check out our tutorial in issue 44 of *The MagPi*: **magpi.cc/44**.

SUBSCRIBE TODAY FROM ONLY £5

SAVE UP TO 35%

MODMYPI
FREE £5
MODMYPI VOUCHER
FOR ALL SUBSCRIBERS

YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
The official Raspberry Pi magazine
Issue 74 | October 2018 | magpi.cc
BUILD A LAPTOP
Digital making on the go with Raspberry Pi
WIN! Media Centre Kits
Haunted Hacks
CreePi projects for Halloween
Code Space Invaders
Recreate this classic in Python
Plus
Grow veggies with the aquaponic garden
Play Rock, paper, scissors with AI
Table football scoreboard
Make a Washing Line Sensor
Keep your clothes dry
New PiCade | Top 10 Cases | QuickStart guide

12 Month Subscription

- **FREE Delivery**
  Get it fast and for FREE
- **Exclusive Offers**
  Great gifts, offers, and discounts
- **Great Savings**
  Save up to 35% compared to stores

Rolling Monthly Subscription

- **Low Monthly Cost** (from £5)
- **Cancel at any time**
  Leave any time applies to Rolling Subscription only
- **Free delivery to your door**
- **Available worldwide**

Subscribe for a Year

£55 (UK)	£90 (USA)
£80 (EU)	£95 (Rest of World)

magpi.cc/subscribe

JOIN FOR 12 MONTHS AND GET A

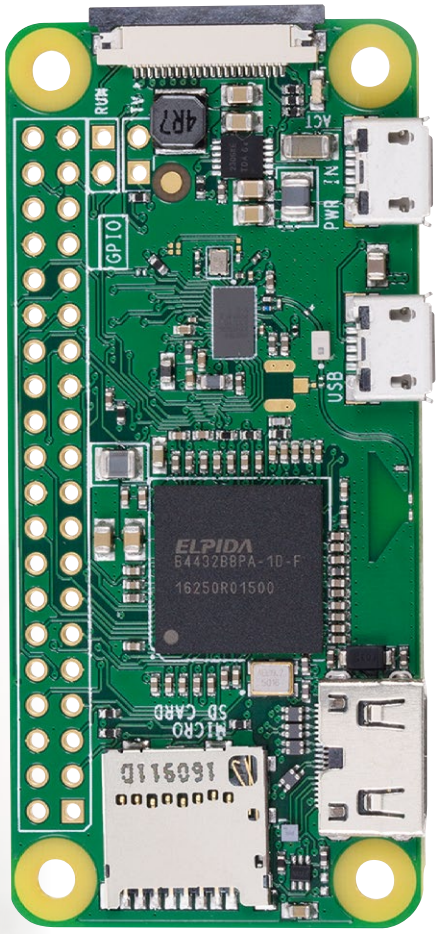
FREE Pi Zero W Starter Kit

WITH YOUR SUBSCRIPTION

Subscribe in print for 12 months today and you'll receive:

WORTH £20

- Pi Zero W
- Pi Zero W case with three covers
- USB and HDMI converter cables
- Camera Module connector



BUY IN PRINT WORLDWIDE! magpi.cc/store
The MagPi
VIDEO GAMES
DESIGN, CREATE & DISTRIBUTE GAMES WITH RASPBERRY PI
LEARN TO CODE BBC TV IS BACK
AIY EDGE REVEALED!
RACE AI CARS WITH FORMULA PI
BUILD A TEENAGE DINNER KLAXON
HOME SECURITY
TOP 10 ROBOTS

magpi.cc/subscribe

SUBSCRIBE on app stores

From **£2.29**

Available on the App Store

GET IT ON Google Play

Part 04

Pygame Zero Invaders

There must be very few people who have not played Space Invaders, and for some it may have been their very first experience of a computer game

MAKER

Mark Vanstone

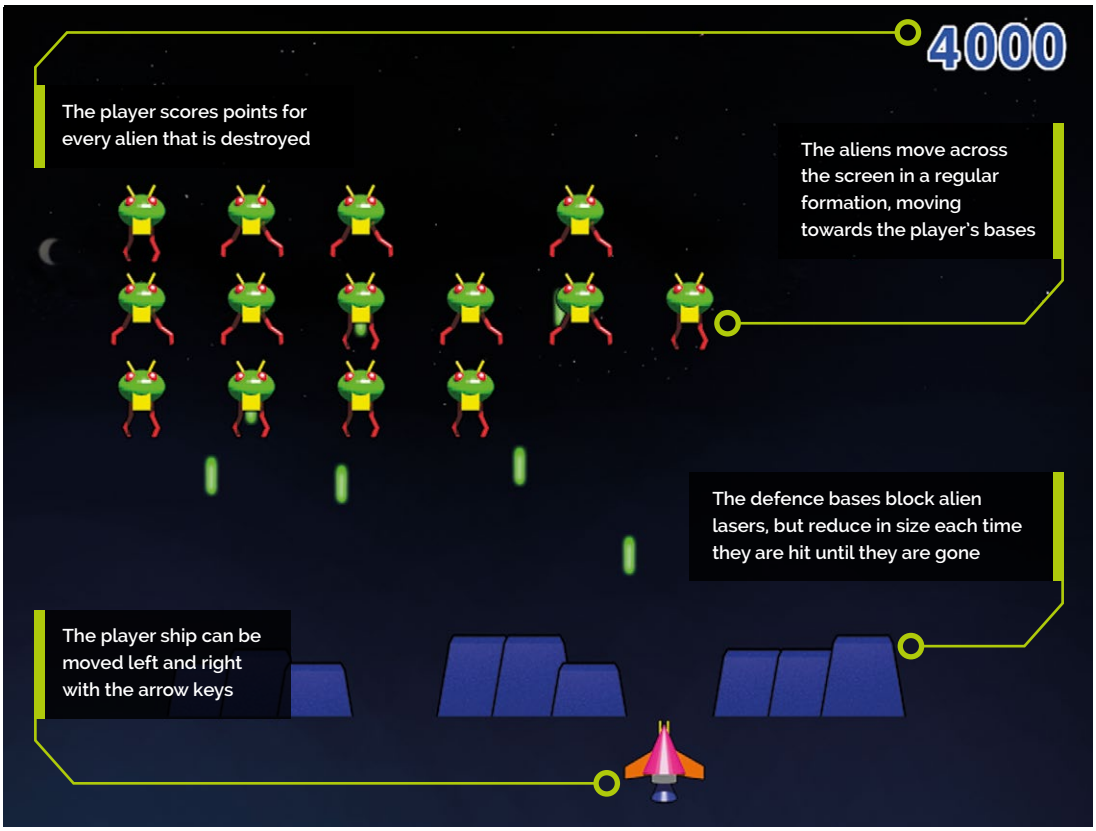
Educational software author from the nineties, author of the ArcVenture series, disappeared into the corporate software wasteland. Rescued by the Raspberry Pi!

magpi.cc/YiZnxl

@mindexplorers

The Space Invaders game format requires quite a few different coding techniques to make it work. For some time, if your author needed to learn a new coding language, he would task himself to write a Space Invaders game in it. This would give a good workout through the syntax and functions of the language.

This tutorial will be split into two parts. In the first we will build a basic invaders game with aliens, lasers, defence bases, and a score. The second part (next issue) will add all the extra bits that make it into the game that appeared in amusement arcades and sports halls in the 1970s.


4000
The player scores points for every alien that is destroyed
The aliens move across the screen in a regular formation, moving towards the player's bases
The defence bases block alien lasers, but reduce in size each time they are hit until they are gone
The player ship can be moved left and right with the arrow keys

You'll Need

- Raspbian Jessie or newer
- An image manipulation program such as GIMP, or images from **magpi.cc/MATfil**
- The latest version of Pygame Zero (1.2)
- A cool head as the lasers rain down on you

01 Let's get stuck in

If you have read the previous episodes of this series, you will know how we set up a basic Pygame Zero program, so we can jump right in to getting things on the screen. We will need some graphics for the various elements of the game – you can design them yourself or use ours from: **magpi.cc/MATfil**. The Pygame Zero default screen size is 800 width by 600 height, which is a good size for this game, so we don't need to define `WIDTH` or `HEIGHT`.

02 A bit of a player

Let's start with getting the player ship on the screen. If we call our graphic **player.png**, then we can create the player Actor near the top of our code by writing `player = Actor("player", (400, 550))`.

We will probably want something a bit more interesting than just a plain black window, so we can add a background in our `draw()` function. If we draw this first, everything else that we draw will be on top of it. We can draw it using the `blit()` function by writing `screen.blit('background', (0, 0))` – assuming we have called our background image **background.png**. Then, to draw the player, just add `player.draw()` afterwards.

03 Let's get moving

We need the player ship to respond to key presses, so we'll check the Pygame Zero keyboard object to see if certain keys are currently pressed. Let's make a new function to deal with these inputs. We will call the function `checkKeys()` and we'll need to call it from our `update()` function.

In the `checkKeys()` function, we write `if keyboard.left:` and then `if player.x > 40: player.x -= 5`. We need to declare the player Actor object as global inside our `checkKeys()` function. We then write a similar piece of code to deal with the right arrow key; **figure1.py** shows how this all fits together.

04 An alien concept

We now want to create a load of aliens in formation. You can have them in whatever format you want, but we'll set up three rows of aliens with six on each row. We have an image called **alien.png** and can make an Actor for each alien that we will store in a list so that we can easily loop through the list to perform actions on them. When we create the alien Actors, we will use a bit of maths to set the initial x and y co-ordinates. It would be a good idea to define a function to set up the aliens – `initAliens()` – and because we will want to set up other elements too, we could define a function `init()`, from which we can call all the setup functions.

05 Doing the maths

To position our aliens and to create them as Actors, we can declare a list – `aliens = []` – and then create a loop using `for a in range(18):`. In this loop, we need to create each Actor and then work out where their x and y co-ordinates will be to start. We can do this in the loop by writing: `aliens.append(Actor("alien1", (210+(a % 6)*80,100+(int(a/6)*64))))`. This may look a little daunting, but we can break it down by saying 'x is 210 plus the remainder of dividing by 6 multiplied by 80'.

This will provide us with x co-ordinates starting at 210 and with a spacing of 80 between each. The y calculation is similar, but we use normal division, make it an integer, and multiply by 64.

figure1.py

```
001. import pgzrun
002.
003. player = Actor("player", (400, 550)) # Load in the player
     Actor image
004.
005. def draw(): # Pygame Zero draw function
006.     screen.blit('background', (0, 0))
007.     player.draw()
008.
009. def update(): # Pygame Zero update function
010.     checkKeys()
011.
012. def checkKeys():
013.     global player
014.     if keyboard.left:
015.         if player.x > 40: player.x -= 5
016.     if keyboard.right:
017.         if player.x < 760: player.x += 5
018.
019. pgzrun.go()
```

▲ Functions to create a player ship and background, display them, and handle moving the player ship

Get The MagPi 71

This is the latest instalment in a series of Pygame Zero tutorials. You can download digital editions of previous tutorials for free. Start with *The MagPi* #71

magpi.cc/71

figure2.py

```
001.  def updateAliens():
002.      global moveSequence, moveDelay
003.      movex = movey = 0
004.      if moveSequence < 10 or moveSequence > 30: movex = -15
005.      if moveSequence == 10 or moveSequence == 30:
006.          movey = 50
007.      if moveSequence >10 and moveSequence < 30: movex = 15
008.      for a in range(len(aliens)):
009.          animate(aliens[a], pos=(aliens[a].x + movex,
      aliens[a].y + movey), duration=0.5, tween='linear')
010.          if randint(0, 1) == 0:
011.              aliens[a].image = "alien1"
012.          else:
013.              aliens[a].image = "alien1b"
014.      moveSequence +=1
015.      if moveSequence == 40: moveSequence = 0
```

▲ The `updateAliens()` function. Calculate the movement for the aliens based on the variable `moveSequence`

06 Believing the strangest things

After that slightly obscure title reference, we shall introduce the idea of the alien having a status. As we have seen in previous instalments, we can add extra data to our Actors, and in this case we will want to add a status variable to the alien after we have created it. We'll explain how we are going to use this a bit later. Now it's time to get the little guys on the screen and ready for action. We can write a simple function called `drawAlien()` and just loop through the alien list to draw them by writing: `for a in range(len(aliens)): aliens[a].draw()` . Call the `drawAlien()` function inside the `draw()` function.

07 The aliens are coming!

We are going to create a function that we call inside our `update()` function that keeps track of what should happen to the aliens. We'll call it `updateAliens()`. We don't want to move the aliens every time the update cycle runs, so we'll keep a counter called moveCounter and increment it each `update()`; then, if it gets to a certain value (`moveDelay`), we will zero the counter. If the counter is zero, we call `updateAliens()`. The `updateAliens()` function will calculate how much they need to move in the x and y directions to get them to go backwards and forwards across the screen and move down when they reach the edges.

08 Updating the aliens

To work out where the aliens should move, we'll make a counter loop from 0 to 40. From 0 to 9 we'll move the aliens left, on 10 we'll move them down, then from 11 to 29 move them right. On 30 they move down and then from 31 to 40 move left. Have a look at **figure2.py** to see how we can do this in the `updateAliens()` function and how that function fits into our `update()` function. Notice how we can use the Pygame Zero function `animate()` to get them to move smoothly. We can also add a switch between images to make their legs move.

09 All your base are belong to us

Now we are going to build our defence bases. There are a few problems to overcome in that we want to construct our bases from Actors, but there are no methods for clipping an Actor when it is displayed. Clipping is a term to describe that we only display a part of the image. This is a method we need if we are going to make the bases shrink as they are hit by alien lasers. What we will have to do is add a function to the Actor, just like we have added extra variables to them before.

10 Build base

We will make three bases which will be made of three Actors each. If we wanted to display the whole image (**base1.png**), we would create a list of base Actors and display each Actor with some code like `bases[0].draw()`. What we want to do is add a variable to the base to show how high we want it to be. We will also need to write a new function to draw the base according to the height variable. Have a look at **figure3.py** to see how we write the new function and attach it to each Actor. This means we can now call this function from each base Actor using: `bases[b].drawClipped()`, as shown in the `drawBases()` function.

11 Can I shoot something now?

To make this into a shooting game, let's add some lasers. We need to fire lasers from the player ship and also from the aliens, but we are going to keep them all in the same list. When we create a new laser by making an Actor and adding it to the

Top Tip

Beware of deleting elements of a list

If you delete a list element while you are looping through it with `range (len(list))`, when you get to the end of the loop it will run out of elements and return an error because the range of the loop is the original length of the list.

list `lasers[]`, we can give the Actor a type. In this case we'll make alien lasers type 0 and player lasers type 1. We'll also need to add a status variable. The creation and updating of the lasers is similar to other elements we've looked at; **figure4.py** (overleaf) shows the functions that we can use.

12 Making the lasers work

You can see in **figure4.py** that we can create a laser from the player by adding a check for the **SPACE** key being pressed in our `checkKeys()` function. We will use the blue laser image called **laser2.png**. Once the new laser is in our list of lasers, it will be drawn to the screen if we call the `drawLasers()` function inside our `draw()` function. In our `updateLasers()` function we loop through the list of lasers and check which type it is. So if it is type 1 (player), we move the laser up the screen and then check to see if it hit anything. Notice the calls to a `listCleanup()` function at the bottom. We will come to this in a bit.

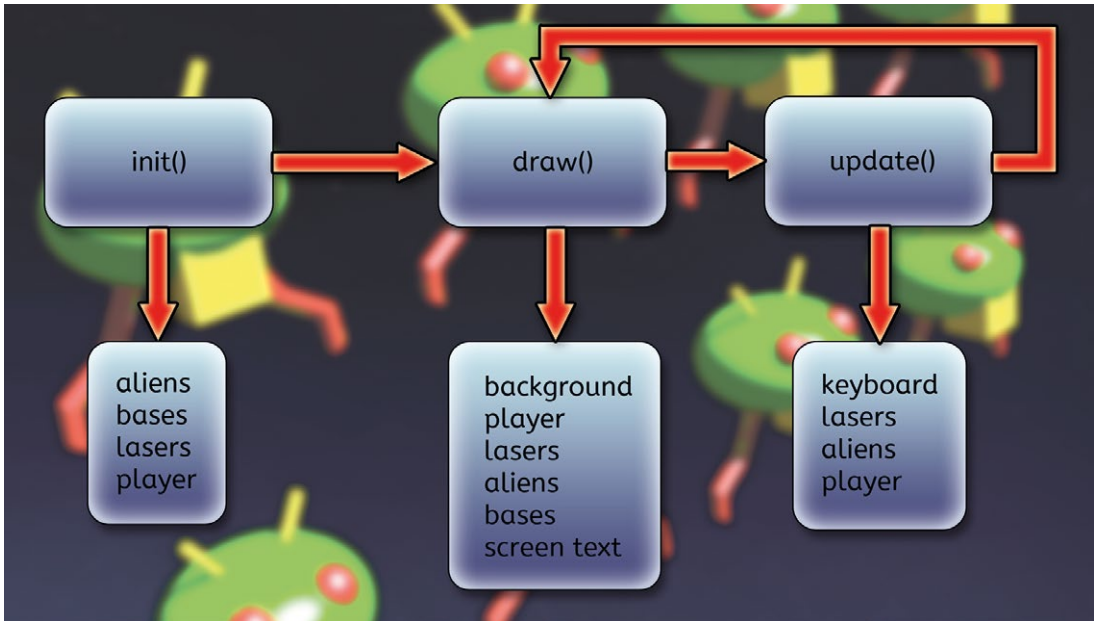
figure3.py

```
001. def drawClipped(self):
002.     screen.surface.blit(self._surf, (self.x-32, self.y-
     self.height+30),(0,0,64,self.height))
003.
004. def initBases():
005.     global bases
006.     bases = []
007.     bc = 0
008.     for b in range(3):
009.         for p in range(3):
010.             bases.append(Actor("base1",
     midbottom=(150+(b*200)+(p*40),520)))
011.             bases[bc].drawClipped = drawClipped.__get__
     (bases[bc])
012.             bases[bc].height = 60
013.             bc +=1
014.
015. def drawBases():
016.     for b in range(len(bases)): bases[b].drawClipped()
```

▲ Setting up an extension function to draw an Actor with clipping

13 Collision course

Let's look at `checkPlayerLaserHit()` first. We can detect if the laser has hit any aliens by looping round the alien list and checking with the Actor function – `collidepoint((lasers[l].x, lasers[l].y))` – to see if a collision has occurred. If an alien has been hit, this is where our status variables come into play. Rather than just removing the laser and the alien from their lists, we need to flag them as ready to remove. The reason for this is that if we remove anything from a list while we are



Top Tip

Write functions for each collective action

To make coding easier to read rather than having lots of code associated with one type of element in the `draw()` or `update()` functions, send it out to a function like `drawLasers()` or `checkKeys()`.

figure4.py

```
001.  def checkKeys():
002.      global player, lasers
003.      if keyboard.space:
004.          l = len(lasers)
005.          lasers.append(Actor("laser2",
      (player.x,player.y-32)))
006.          lasers[l].status = 0
007.          lasers[l].type = 1
008.
009.  def drawLasers():
010.      for l in range(len(lasers)): lasers[l].draw()
011.
012.  def updateLasers():
013.      global lasers, aliens
014.      for l in range(len(lasers)):
015.          if lasers[l].type == 0:
016.              lasers[l].y += (2*DIFFICULTY)
017.              checkLaserHit(l)
018.              if lasers[l].y > 600: lasers[l].status = 1
019.          if lasers[l].type == 1:
020.              lasers[l].y -= 5
021.              checkPlayerLaserHit(l)
022.              if lasers[l].y < 10: lasers[l].status = 1
023.      lasers = listCleanup(lasers)
024.      aliens = listCleanup(aliens)
```

▲ Checking the keys that are pressed, creating lasers, moving them, and checking if they have collided with anything

looping through any of the lists then by the time we get to the end of the list, we are an element short and an error will be created. So we set these Actors to be removed with `status` and then remove them afterwards with `listCleanup()`.

Top Tip

Collect all your setup code in one place

If possible, it is good to have as much of the code that sets everything back to the beginning in one place so that you can easily restart the game.

14 Cleaning up the mess

The `listCleanup()` function creates a new empty list, then runs through the list that is passed to it, only transferring items to the new list that have a status of 0. This new list is then returned back and used as the list going forward. Now that we have made a system for one type of laser we can easily adapt that for our alien laser type. We can create the alien lasers in the same way as the player lasers, but instead of waiting for a keyboard press we can just produce them at random intervals using `if randint(0, 5) == 0:` when we are updating our aliens. We set the type to 0 rather than 1 and move them down the screen in our `updateLasers()` function.

15 Covering the bases

So far, we haven't looked at what happens when a laser hits one of the defence bases. Because we are changing the height of the base Actors, the built-in collision detection won't give us the result we want, so we need to write another custom function to check laser collision on the base Actor. Our new function, `collideLaser()` will check the laser co-ordinates against the base's co-ordinates, taking into account the height of the base. We then attach the new function to our base Actor when it is created. We can use the new `collideLaser()` function for checking both the player and the alien lasers and remove the laser if it hits – and if it is an alien laser, reduce the height of the base that was hit.

16 Laser overkill

We may want to change the number of lasers being fired by the aliens, but at the moment our player ship gets to fire a laser every `update()` cycle. If the **SPACE** key is held down, a constant stream of lasers will be fired, which not only is a little bit unfair on the poor aliens but will also take its toll on the speed of the game. So we need to put some limits on the firing speed and we can do this with another built-in Pygame Zero object: the clock. If we add a variable `laserActive` to our player Actor and set it to zero when it fires, we can then call `clock.schedule(makeLaserActive, 1.0)` to call the function `makeLaserActive()` after 1 second.

17 I'm hit! I'm hit!

We need to look now at what happens when the player ship is hit by a laser. For this we will make a multi-frame animation. We have five explosion images to put into a list, with our normal ship image at the beginning, and attach it to our player Actor. We need to import the Math module, then in each `draw()` cycle we write: `player.image = player.images[math.floor(player.status/6)]`, which will display the normal ship image while `player.status` is 0. If we set it to 1 when the player ship is hit, we can start the animation in motion. In the `update()` function we write: `if player.status > 0: player.status += 1`. As the status value increases, it will start to draw the sequence of frames one after the other.

18 Initialisation

Now, it may seem a bit strange to be dealing with initialisation near the end of the tutorial, but we have been adding and changing the structure of our game elements as we have gone along and only now can we really see all the data that we need to set up before the game starts. In Step 04 we created a function called `init()` that we should call to get the game started. We could also use this function to reset everything back to start the game again. If we have included all the initialisation functions and variables we have talked about, we should have something like **figure5.py**.

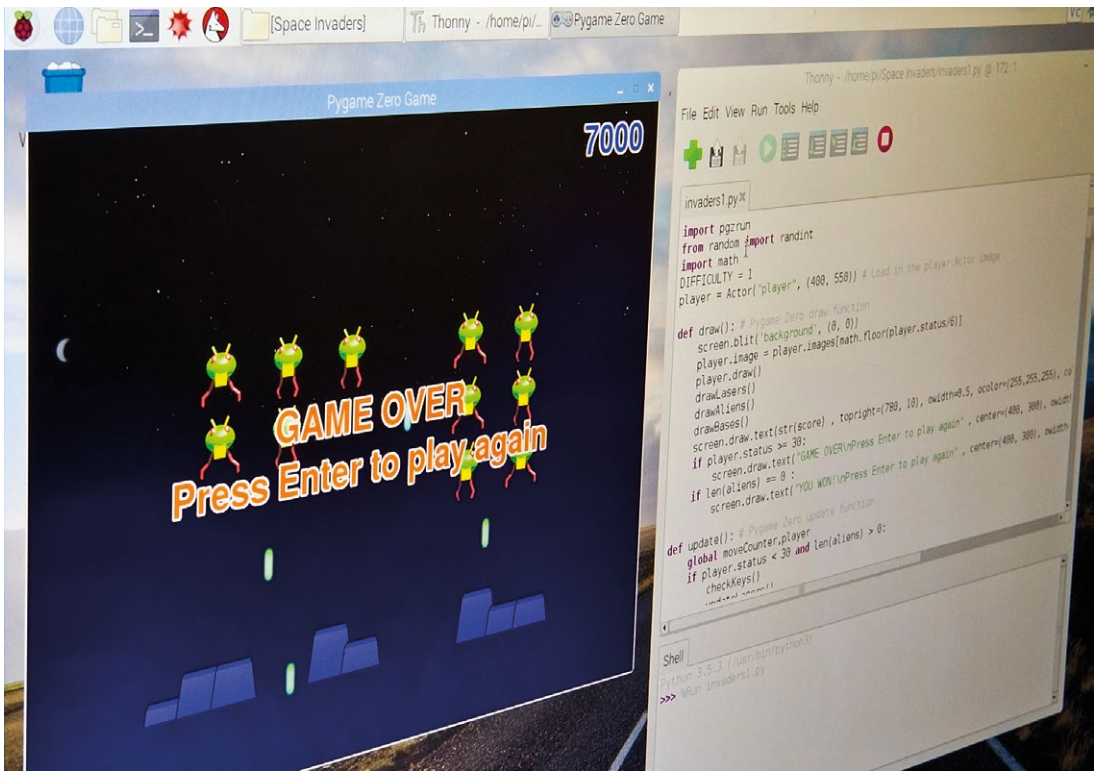
figure5.py

```
001. def init():
002.     global lasers, score, player, moveSequence,
     moveCounter, moveDelay
003.     initAliens()
004.     initBases()
005.     moveCounter = moveSequence = player.status = score =
     player.laserCountdown = 0
006.     lasers = []
007.     moveDelay = 30
008.     player.images = ["player","explosion1","explosion2",
     "explosion3","explosion4","explosion5"]
009.     player.laserActive = 1
```

▲ The initialisation of our data. Calling this function sets our variables back to their start values

19 They're coming in too fast!

There are a few finishing touches to do to complete this first part. We can set a `DIFFICULTY` value near the top of the code and use it on various elements to make the game harder. We should also add a score, which we do by adding 1000 to a global variable score if an alien is hit, and then display that in the top right of the screen in the `draw()` function. When the game finishes (the player has been hit or all the aliens are gone), we should display a suitable message. Have a look at the complete listing to see how these bits fit in. When that's all done, we should have the basis of a Space Invaders game. In the next part of this series we will add more into the game, such as levels, lives, sound, bonus aliens, and a leaderboard.



◀ It's game over for now, but we'll be back next issue to improve the game

Top Tip

Define several variables at once

If you are setting several variables to the same value, you can combine them into one line by writing `a = b = c = 0` to set a, b, and c to zero.

invaderspart1.py

```
import pgzrun
from random import randint
import math
DIFFICULTY = 1
player = Actor("player", (400, 550)) # Load in the player Actor image

def draw(): # Pygame Zero draw function
    screen.blit('background', (0, 0))
    player.image = player.images[math.floor(player.status/6)]
    player.draw()
    drawLasers()
    drawAliens()
    drawBases()
    screen.draw.text(str(score), topright=(780, 10), owidth=0.5, ocolor=(255,255,255), color=(0,64,255), fontsize=60)
    if player.status >= 30:
        screen.draw.text("GAME OVER\nPress Enter to play again" , center=(400, 300), owidth=0.5, ocolor=(255,255,255), color=(255,64,0), fontsize=60)
    if len(aliens) == 0 :
        screen.draw.text("YOU WON!\nPress Enter to play again" , center=(400, 300), owidth=0.5, ocolor=(255,255,255), color=(255,64,0) , fontsize=60)

def update(): # Pygame Zero update function
    global moveCounter,player
    if player.status < 30 and len(aliens) > 0:
        checkKeys()
        updateLasers()
        moveCounter += 1
        if moveCounter == moveDelay:
            moveCounter = 0
            updateAliens()
        if player.status > 0: player.status += 1
    else:
        if keyboard.RETURN: init()

def drawAliens():
    for a in range(len(aliens)): aliens[a].draw()

def drawBases():
    for b in range(len(bases)):
        bases[b].drawClipped()

def drawLasers():
    for l in range(len(lasers)): lasers[l].draw()

def checkKeys():
    global player, lasers
    if keyboard.left:
        if player.x > 40: player.x -= 5
    if keyboard.right:
        if player.x < 760: player.x += 5
    if keyboard.space:
        if player.laserActive == 1:
            player.laserActive = 0
            clock.schedule(makeLaserActive, 1.0)
            l = len(lasers)
            lasers.append(Actor("laser2", (player.x,player.y-32)))
            lasers[l].status = 0
            lasers[l].type = 1

def makeLaserActive():
    global player
    player.laserActive = 1

def checkBases():
    for b in range(len(bases)):
        if l < len(bases):
            if bases[b].height < 5:
                del bases[b]

def updateLasers():
    global lasers, aliens
    for l in range(len(lasers)):
        if lasers[l].type == 0:
            lasers[l].y += (2*DIFFICULTY)
            checkLaserHit(l)
            if lasers[l].y > 600:
                lasers[l].status = 1
        if lasers[l].type == 1:
            lasers[l].y -= 5
            checkPlayerLaserHit(l)
            if lasers[l].y < 10:
                lasers[l].status = 1
    lasers = listCleanup(lasers)
    aliens = listCleanup(aliens)

def listCleanup(l):
    newList = []
    for i in range(len(l)):
        if l[i].status == 0: newList.append(l[i])
    return newList

def checkLaserHit(l):
    global player
```

DOWNLOAD THE FULL CODE:
magpi.cc/lwqLZj

```
092.     if player.collidepoint((lasers[l].x,
    lasers[l].y)):
093.         player.status = 1
094.         lasers[l].status = 1
095.     for b in range(len(bases)):
096.         if bases[b].collideLaser(lasers[l]):
097.             bases[b].height -= 10
098.             lasers[l].status = 1
099.
100. def checkPlayerLaserHit(l):
101.     global score
102.     for b in range(len(bases)):
103.         if bases[b].collideLaser(lasers[l]):
104.             lasers[l].status = 1
105.     for a in range(len(aliens)):
106.         if aliens[a].collidepoint((lasers[l].x,
    lasers[l].y)):
107.             lasers[l].status = 1
108.             aliens[a].status = 1
109.             score += 1000
110.
111. def updateAliens():
112.     global moveSequence, lasers, moveDelay
113.     movex = movey = 0
114.     if moveSequence < 10 or moveSequence > 30:
115.         movex = -15
116.     if moveSequence == 10 or moveSequence == 30:
117.         movey = 50 + (10 * DIFFICULTY)
118.         moveDelay -= 1
119.     if moveSequence >10 and moveSequence < 30:
120.         movex = 15
121.     for a in range(len(aliens)):
122.         animate(aliens[a], pos=(aliens[a].x + movex,
    aliens[a].y + movey), duration=0.5, tween='linear')
123.         if randint(0, 1) == 0:
124.             aliens[a].image = "alien1"
125.         else:
126.             aliens[a].image = "alien1b"
127.             if randint(0, 5) == 0:
128.                 lasers.append(Actor("laser1",
    (aliens[a].x,aliens[a].y)))
129.                 lasers[len(lasers)-1].status = 0
130.                 lasers[len(lasers)-1].type = 0
131.         if aliens[a].y > 500 and player.status ==
    0:
132.             player.status = 1
133.     moveSequence +=1
134.     if moveSequence == 40: moveSequence = 0
135.
136. def init():
137.     global lasers, score, player, moveSequence,
    moveCounter, moveDelay
138.     initAliens()
139.     initBases()
140.     moveCounter = moveSequence = player.status =
    score = player.laserCountdown = 0
141.     lasers = []
142.     moveDelay = 30
143.     player.images =
    ["player","explosion1","explosion2",
    "explosion3","explosion4","explosion5"]
144.     player.laserActive = 1
145.
146. def initAliens():
147.     global aliens
148.     aliens = []
149.     for a in range(18):
150.         aliens.append(Actor("alien1", (210+
151. (a % 6)*80,100+(int(a/6)*64))))
152.         aliens[a].status = 0
153.
154. def drawClipped(self):
155.     screen.surface.blit(self._surf, (self.x-32,
    self.y-self.height+30),(0,0,64,self.height))
156.
157. def collideLaser(self, other):
158.     return (
159.         self.x-20 < other.x+5 and
160.         self.y-self.height+30 < other.y and
161.         self.x+32 > other.x+5 and
162.         self.y-self.height+30 + self.height >
    other.y
163.     )
164.
165. def initBases():
166.     global bases
167.     bases = []
168.     bc = 0
169.     for b in range(3):
170.         for p in range(3):
171.             bases.append(Actor("base1",
    midbottom=(150+(b*200)+(p*40),520)))
172.             bases[bc].drawClipped =
    drawClipped.__get__(bases[bc])
173.             bases[bc].collideLaser =
    collideLaser.__get__(bases[bc])
174.             bases[bc].height = 60
175.             bc +=1
176.
177. init()
178. pgzrun.go()
```

Part 02

Pi Bakery: The Matrix

The Matrix is an undedicated array of switches and lights that can be put to any number of uses. Part 2 of our tutorial shows you how to make the controller electronics

MAKER

Mike Cook

Veteran magazine author from the old days, writer of the Body Build series, plus co-author of *Raspberry Pi for Dummies*, *Raspberry Pi Projects*, and *Raspberry Pi Projects for Dummies*.

magpi.cc/259aT3X

Last month we showed you how to modify the cheap, white LED stick-on battery powered lights that are around at the moment in bargain shops. This month we will show you how to mount these light modules in a box and also detail the controller board needed to drive them. When completed, this matrix board can be used for many things, such as triggering sounds from Sonic Pi, making animated light shows, and creating interactive games. It is one project with many uses, and is a perfect introduction to get you writing your own unique code.

01 Removing the stickies

We need to remove the bits of the light assembly we don't need. First off, peel off the CE stickers off the side of the light's cover. Next, remove the double-sided self-adhesive pads off the base. This is easily done by peeling back a small part of the pad's covering and getting your fingernail underneath the pad. Then, with a rolling action of the thumb, push the pad off the plastic. The adhesive layer should come off the plastic and stay on the pad, as shown in **Figure 1**. Finally, use a pair of fine-nose pliers and pull out the two battery clips in the tray that did not have wires attached to them. Leave the ones that have soldered wires as a reference point.

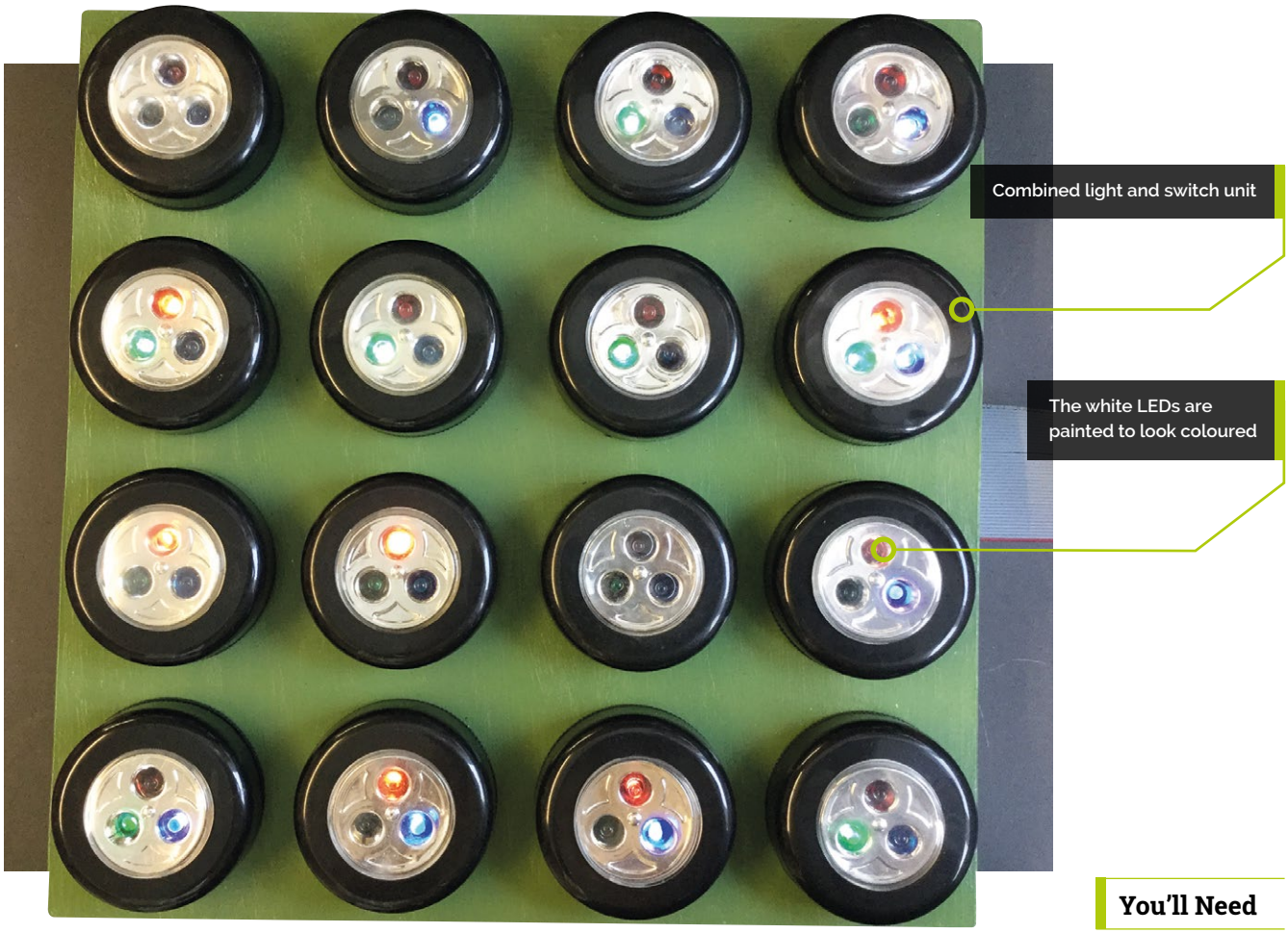
Figure 1

Figure 1 Rolling off the sticky foam pads

Top Tip

Using a Dremel

We would recommend getting the flexible drive extension if you are using a Dremel, as this gives you a smaller and much more manoeuvrable working head.



02 Making the exit slot

Use a handheld drill with a router bit – we used a 4.8 mm high-speed cutter Dremel bit – and cut a slot in the right-hand side of the battery tray. Make sure the slot goes all the way down to the bottom of the tray. **Figure 2** shows the slot part of the way to the base of the tray. Do this for 15 of the 16 battery trays. On the last one, cut this slot in the side opposite remaining clips. This is for position 11 and needs to be at another place to allow the wires to exit clear of the control board when it is fitted. Clean up the swarf with a scalpel and vacuum cleaner.

03 Drilling fixing holes

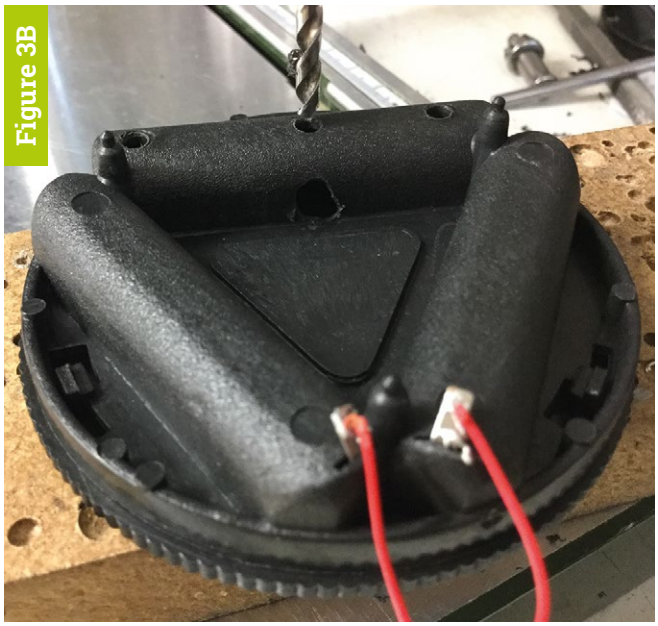
Take the battery tray, and attach it to the base, then drill three 3 mm holes through the tray and the base. Two fixing holes should be drilled where there is a mould mark at the opposite end of the battery housings with the clips still attached. Make sure you get the drill as close to the centre of this mark as possible. The third hole is for the exit wires and its position is not very critical. It should be drilled through the battery housing close to where you created the slot. Note that this will be in a different position for the position 11 housing. **Figure 3** (overleaf) shows the drilling, for position 11 on the left (**3A**), and all other positions on the right (**3B**).

Figure 2

Figure 2 Cutting the wire exit hole

You'll Need

- MAX7219 LED display driver **magpi.cc/iVpBxJ**
- 74HTC14 inverter **magpi.cc/eKazXA**
- 40-way 2-row pin header **magpi.cc/QtLcoM**
- Stripboard 64×95 mm **magpi.cc/CaKrVk**
- 16 × 1 kΩ, ⅛ W resistors
- 1 × 33 kΩ, ⅛ W resistor
- 1 × 0.1 µF ceramic capacitor
- 1 × 68 0µF 10 V electrolytic capacitor
- 32 × 10 mm M3 pan-head screws
- 32 × M3 nuts



▲ **Figure 3**
Left – position 11 drillings, Right – all other positions

04 Removing ridges

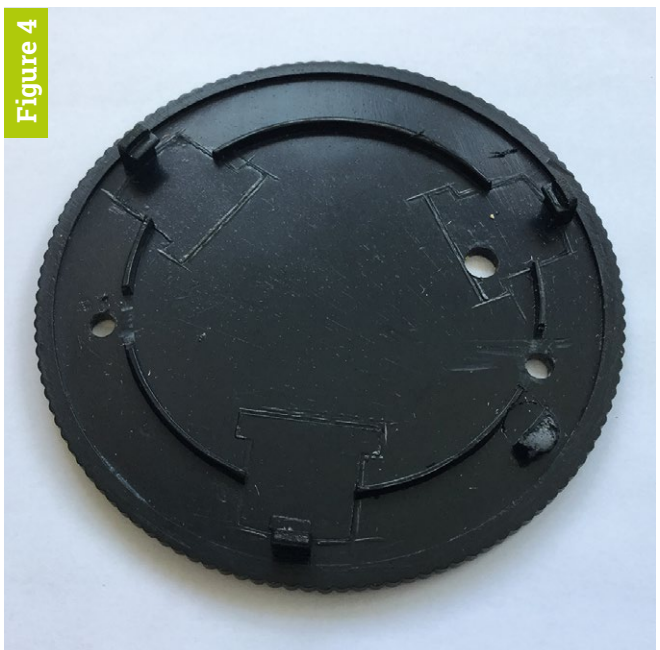
Remove the base from the battery tray and clean up the holes with a scalpel, and again remove the swarf with a vacuum cleaner. Then find the wire exit hole on the base and enlarge it to a 4 mm diameter hole. Now the two fixing holes will be close to a small ridge on the base. This prevents the pan-head screw from sitting flat, so, using a scalpel, cut off a small amount of this ridge so it can sit flat. **Figure 4** shows the ridges removed.

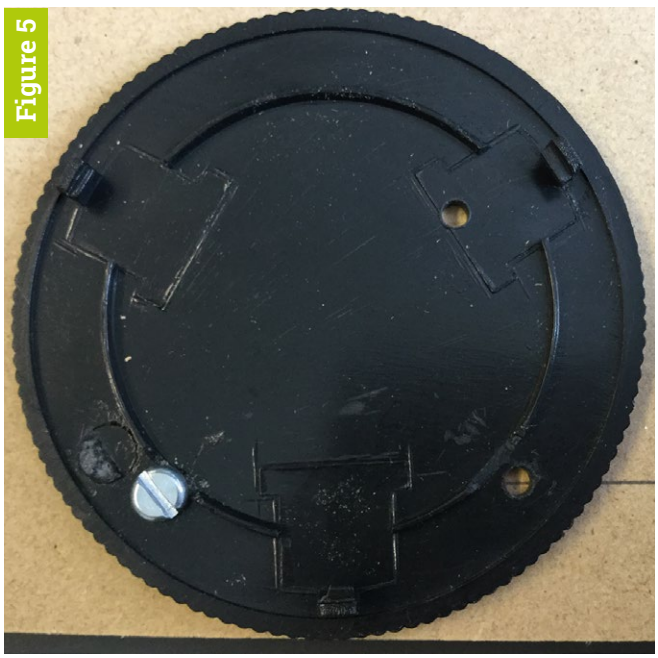
05 Aligning fixing holes

Fix the left-hand hole of the base to the left-hand hole you have already drilled in the top of the box with an M3 nut and bolt. Now rotate the base until you see the line you have drawn on the top through the right-hand hole in the base and tighten up the nut. Then drill a 3 mm hole through the top, using the existing hole in the base as a template. This way, all the holes will line up. **Figure 5** shows the line through the hole.

▼ **Figure 4**
Removing the small plastic ridges round the mounting holes

▶ **Figure 5**
With one mounting screw fastened, rotate the base until you can see the line through the hole. Then drill through, using the base as a template





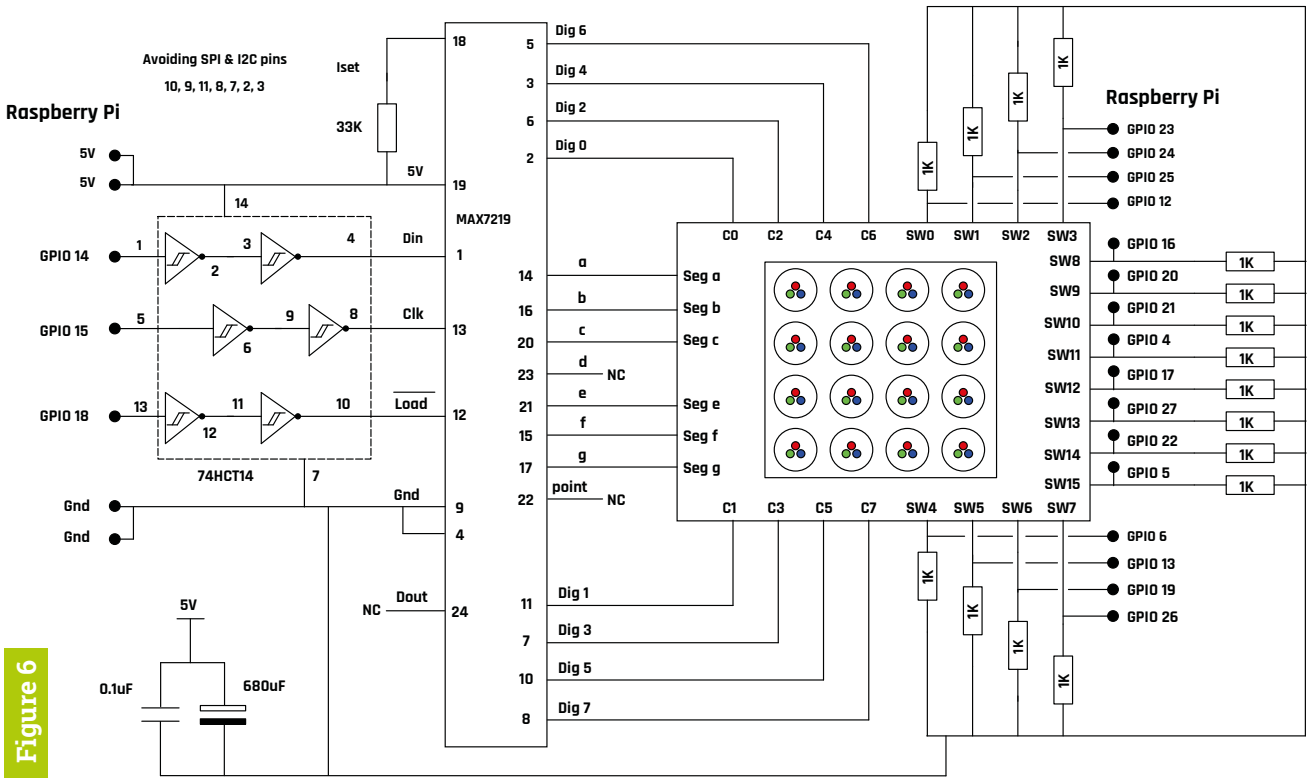


Figure 6

▲ **Figure 6** The controller board schematic

06 Fitting the bases

Now fasten the base through the newly drilled holes with another M3 nut and bolt, and drill a 4 mm wire exit hole through the box top, using the remaining hole in the base as a template guide. At this point it is best to number the bases with the position number so you always use the same set of holes when fixing the bases to the board. We used some sticky paper labels cut up small to do this. Now remove all the bases from the board and put them to one side, as it is time to paint the box.

07 Painting the box

Sand any pencil marks from the top of the box and apply two coats of MDF primer, sanding between coats. Then choose your paint colour and apply two coats of the recommended undercoats, again sanding between coats. Then you will probably only need one layer of top coat to make it look good. Try to choose paints that all clean up with water: we find that these sorts are more quickly recoatable. You only need to paint the top and sides; stand the box on four props to prevent paint sticking. We used four pots of transparent paint for this.

08 Understanding the control board

Figure 6 shows the schematic of the control board, which contains just two integrated circuits. The MAX7219 chip does all the work of multiplexing, and the 74HCT14 interfaces between the 3.3 V output signals of the Pi and the 5 V signals needed to control the MAX7219. The 33 kΩ resistor between pins 18 and 19 controls the overall brightness of the LEDs. The large square on the right-hand side is the matrix of LEDs and switches. This is another schematic in its own right and will be shown next month. This method of showing sub-blocks in a schematic, which in themselves are schematics, is known as a hierarchical diagram and makes complex schematics easer to read. We shall see inside that block next month.

> "The MAX7219 chip does all the work of multiplexing, and the 74HCT14 interfaces between the 3.3 V output signals of the Pi and the 5 V signals needed to control the MAX7219"

Figure 7

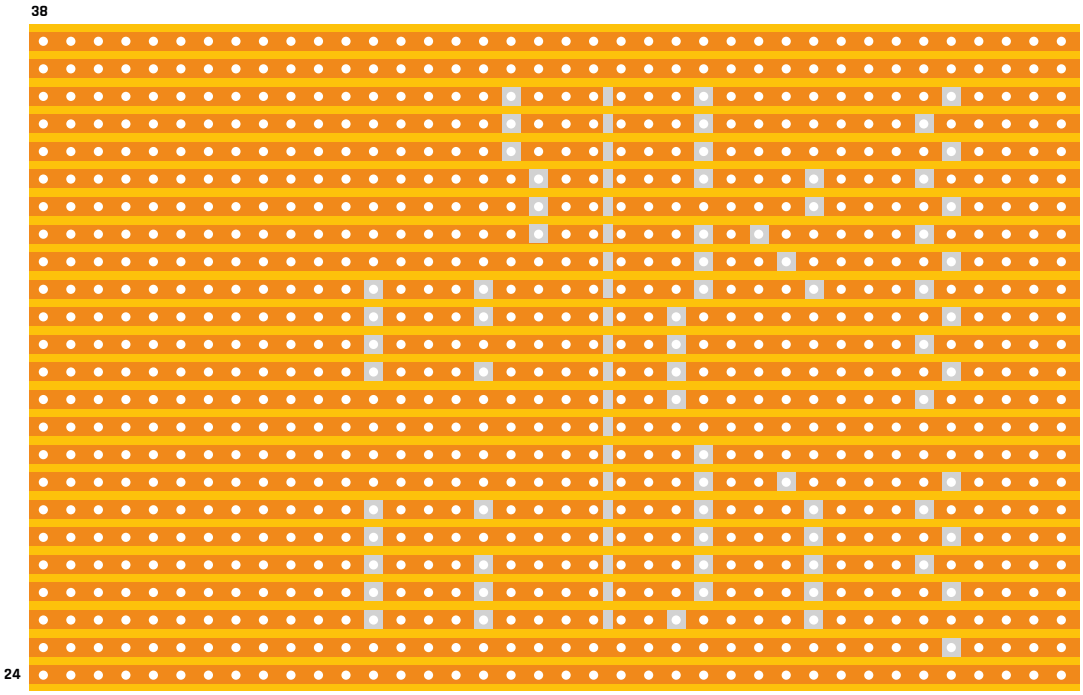
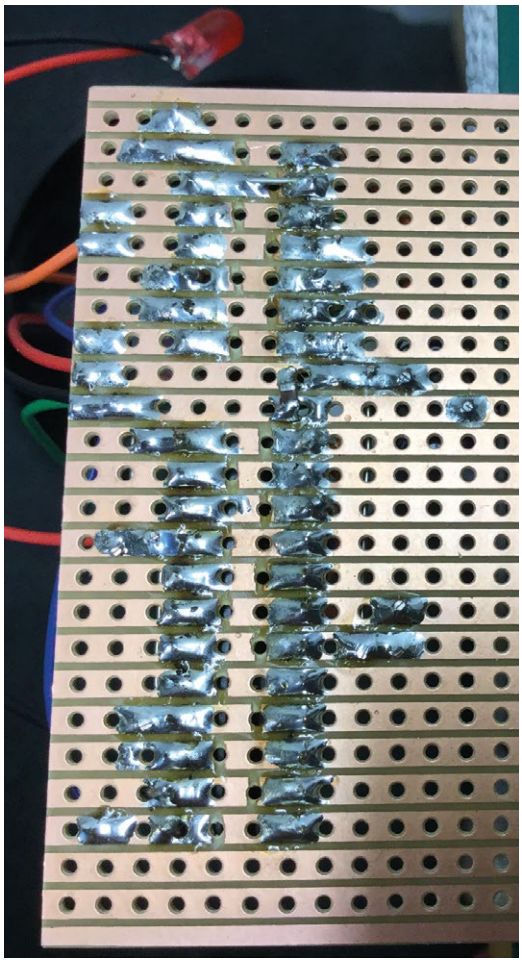


Figure 7
The cuts to make in the stripboard tracks



Make the breaks between holes by cutting two lines close together and then gouging out the copper in between

09 Understanding the pin assignment

What pins to connect to what devices is an important part of any design. The MAX7219 chip requires a 16-bit SPI word which can not be produced by the normal SPI bus, so we opted to 'bit-bang' the protocol. This means specifically setting pins high and low in software, in order to generate the required protocol. As such, we could have used any of the GPIO pins to do this, so we arbitrarily picked pins 14, 15, and 18. The switches in the light clusters likewise could be assigned to any pin, but we choose a bunch of multifunction pins to avoid, mainly the I^2C and SPI pins, to allow other things to be connected alongside the matrix.

10 Preparing the stripboard

Take a piece of stripboard, 24 rows by 38 holes in size, and cut the tracks indicated by the grey area, on the underside in the pattern of **Figure 7**. There are two types of breaks here. First, there's the type centred on a hole. These can be made with a drill bit turned by hand – or spot face cutter, a tool especially made for the job. Or you can make them with a scalpel by scooping out the thin copper tracks above and below the hole. The other type are breaks between the holes, which we prefer to make with a scalpel, although you can use a Dremel and cutting disc if you want.

Top Tip

Using scalpels

While scalpels are ideal, being very sharp and making clean cuts, always make sure you cut away from you – otherwise a slip could result in a nasty cut.

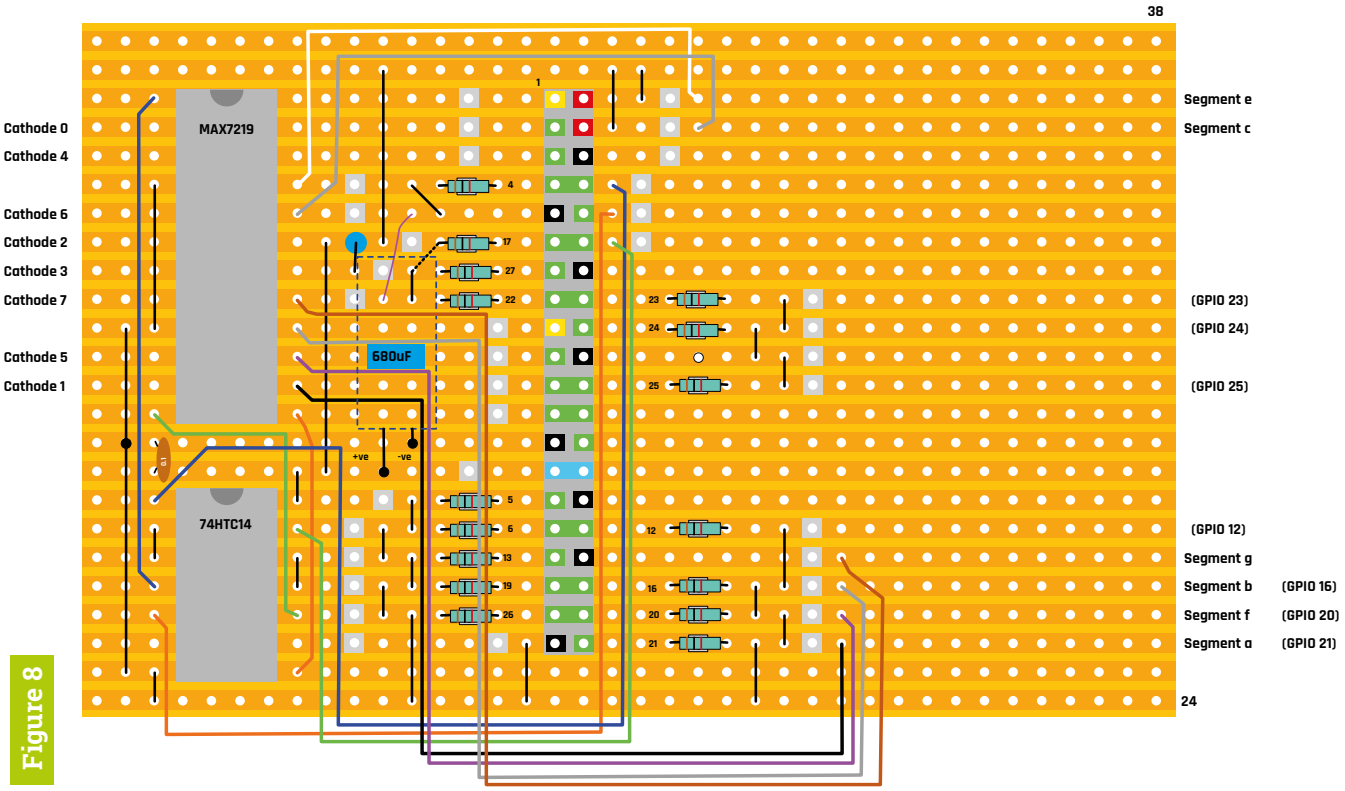


Figure 8

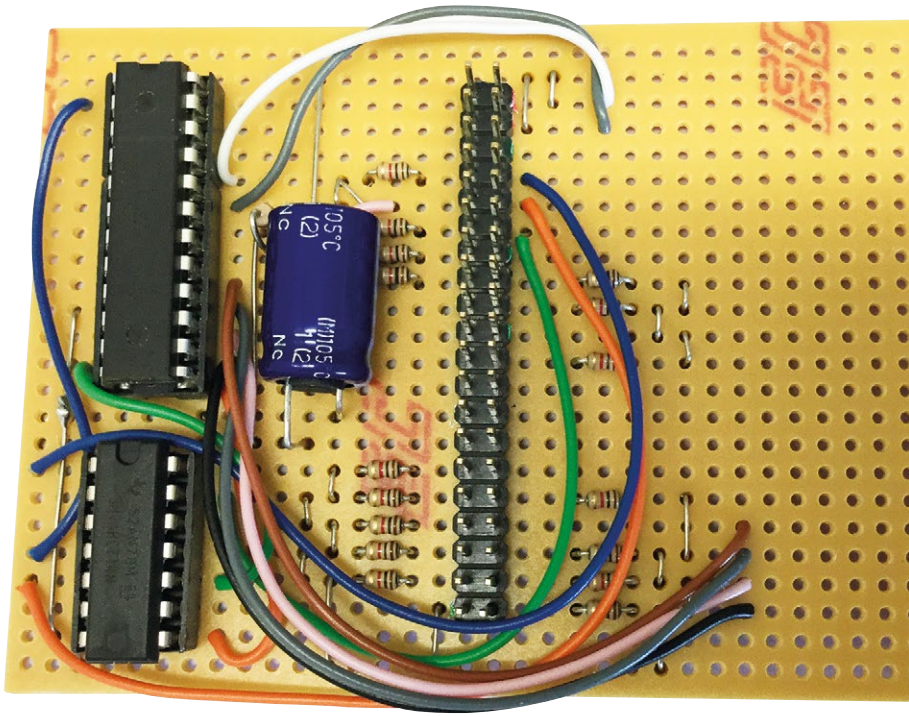
▲ **Figure 8** The physical layout of the controller board

11 Building the board

See **Figure 8** above. We like to use IC sockets for the two chips, although you can make it without. For the MAX7219 we used an 8-pin socket next to a 16-pin socket to make it up to 24 pins. For the 40-way pin header connector we recommend the coloured one from Pimoroni, because it made locating the wires easer. The numbers next to the connector are the GPIO numbers corresponding to that pin. The 1 kΩ switch resistors should be ¼ W so they fit neatly, spanning just one hole. The 680 µF capacitor is shown as a dotted outline, so you can see the details underneath, and should be fitted last.

12 Further construction notes

The tracks and breaks are on the underside, but they are shown as dotted hidden detail, as per normal drawing practice. You will notice that one of the wire links continues in a diagonal line after it goes through the board, on its way to GPIO 17. The labels around the outside of the board are a handy reference for when we complete the final stage of wiring the matrix light/switch units to the control board.



▲ A photograph of the control board

Well, that wraps it up for this month. Next time we will see how to wire the matrix to the control board, and what software we need to bring the matrix to life.

Laundry-saving Rain Detector

Save your washing from a soaking. This easy-to-build wire-free rain detector alerts you to sudden downpours

MAKER

PJ Evans

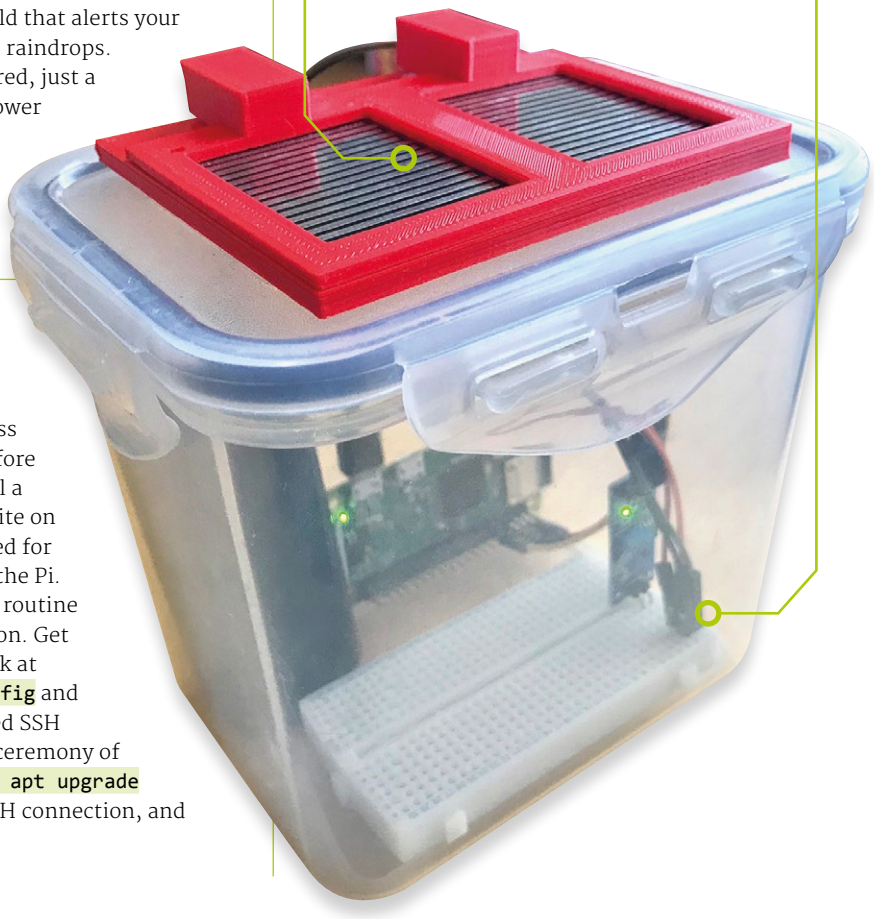
PJ is a software engineer and tinkerer who has littered his house with Raspberry Pi devices. He mostly has nice dry clothes.

mrpjevans.com

@mrpjevans

There's nothing quite like clean air-dried clothes fresh from the line, unless an unexpected shower ruins everything. Ever heard that cry of "RAIN!" from a member of the household, only to be followed by the thundering of feet down the stairs in a desperate bid to save your Sunday best from another trip to the washing machine?

Catch the rain as soon as it starts with this simple standalone build that alerts your phone as soon as it detects raindrops. There's no soldering required, just a few cables. We need low power consumption and WiFi, so this is a perfect project for a Raspberry Pi Zero W.


The Pi, controller, and power are kept safe in the airtight box
The rain sensor works by water shorting the connection. Two are used to increase surface area

You'll Need

- 2 × Rain sensor boards with one controller **magpi.cc/pMUaWu**
- Small breadboard
- Small USB power bank e.g. **magpi.cc/iYvwEL**
- Airtight small food container
- Jumper cables

01 Prepare the Pi

When everything is assembled, it may be tricky later on to gain access to the Raspberry Pi. So, before doing anything else, install a copy of Raspbian Stretch Lite on an SD card (we have no need for a desktop) and insert into the Pi. It's now time for the usual routine of updates and configuration. Get the Pi on your WiFi network at this point using `raspi-config` and make sure you have enabled SSH access. Perform the usual ceremony of `sudo apt update && sudo apt upgrade` then reboot, check your SSH connection, and then power down.

▲ Here's everything you need to build your rain detector. Try to use a container with a rubber or silicone seal

02 Mount the sensors to the lid

You can use any number of sensors you wish, but two works well. Either secure the two plates to the lid using insulation or duct tape. Alternatively, 3D-print the enclosure pictured (STL files available from **magpi.cc/DAuqUT**) and secure with glue or sticky pads.

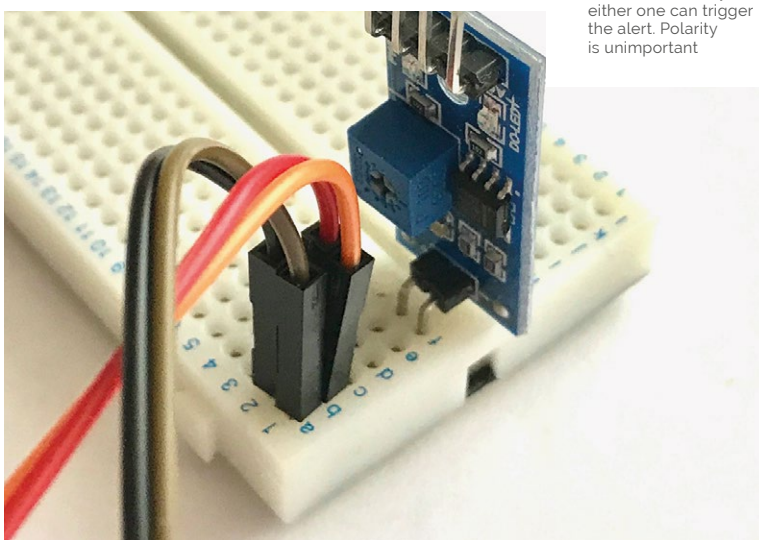
Two pairs of jumper cables need to be attached; one to each sensor plate. Polarity does not matter. The other end of the cables will have to thread into the container, so make as small a hole as possible in an appropriate place so the wires can get through, minimising the chance of any water ingress.

03 Connect the sensors to the controller

In order for the Raspberry Pi to understand what's going on, a small controller board (supplied with the sensors) is required. This takes the small current that is shorted by water and converts it into a digital signal. Using the breadboard, connect the two pairs of wires from the sensors in parallel (so that either sensor could make the circuit) and then insert the controller's receiving pins (the side with two connectors) into the breadboard so that each pin connects with one wire from each sensor.

04 Connecting the controller

To complete our circuit, take a careful look at the four pins on the controller board. They will be marked as A0, D0, GND, and VCC. Using some jumper wires, hook the controller up to the Pi as follows: VCC to GPIO pin 2 (5V), GND to any GND on the GPIO (e.g. pin 6) and D0 to GPIO 17 (pin 11). D0 and A0 are two different ways of reading output from the sensor. D0 is a straight digital on or off, the threshold being controlled by the variable resistor on the board. A0 is an analogue output that (when converted to digital) ranges between 0 and 1024 depending on how heavy the rain is.



▼ The sensor plates need to be connected to the controller board in parallel, not in series. That way, either one can trigger the alert. Polarity is unimportant

Top Tip

Nobody wants a soggy Pi

A critical part of this project is ensuring that, in the event of a downpour, your precious Pi isn't ruined by rain. Make sure the hole for the wires is small and covered.

Top Tip

Not keen on Pushover?

The `pushover()` function in the code can be replaced with anything you like. It could send an email, ping a website, send a text message, or launch a firework (hint: bad idea).

05 Assembly

Connect the micro USB cable from your power bank to the power input on the Raspberry Pi and arrange everything inside your container. Ideally things shouldn't move about, so keep everything in place with sticky pads or tack. You should now be able to seal the container with everything inside, the wires to the sensor plates coming out without being squashed or stressed. Once you're happy, open it up and attach the power bank, then close it again and check your connection. The power bank, depending on its rating, should keep the Pi Zero W alive for a few hours at least.

06 Software

Add the script printed here and save it as **rainbot.py** (or download from GitHub) into a convenient spot such as **~/pi/rainbot**. Once in place, perform an initial test by running `python3 ~/pi/rainbot/rainbot.py`. You should see a readout every five seconds: 'True' if it's dry, 'False' if it's wet. Press **CTRL+C** to stop the script.

07 Pushover

To get alerts, we're going to use Pushover, a neat one-time-payment notification service for smartphones (there's a seven-day free trial). Sign up at **pushover.net**. When logged in, you'll see a 'User Key'; make a copy of this. Now follow the instructions to create an 'Application Token'. You'll be given an API key. Edit the script to replace the API key values, where prompted, with the keys you have been given. Make sure the Pushover app is installed on your phone.

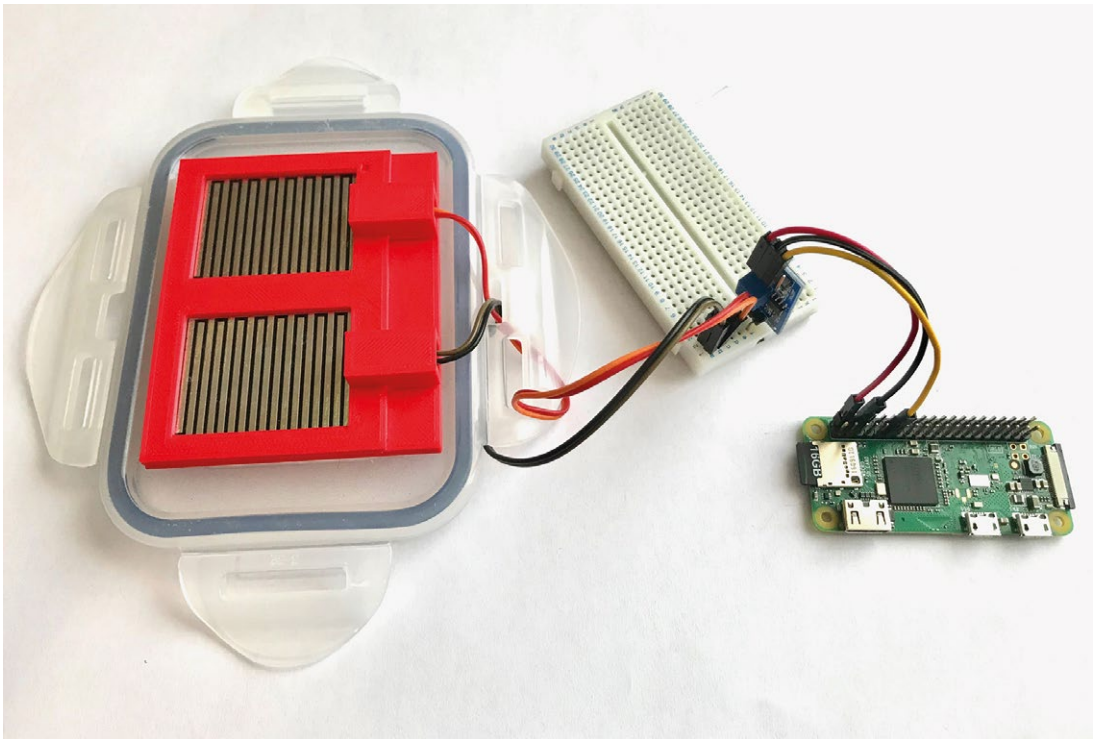
Run the script again. This time, wet one of the panels slightly. A light should illuminate on the controller. If all is well, a few seconds later your phone will display an alert.

08 Run automatically

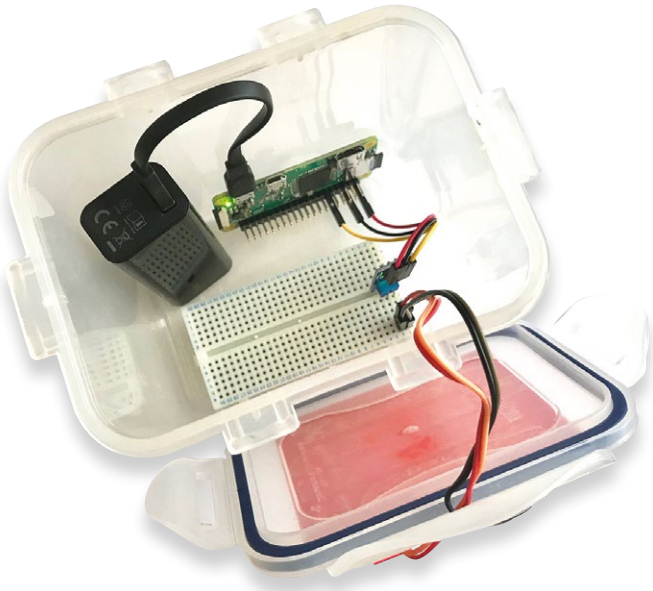
Let's set the script to run on startup.

Create the following file as a superuser:

```
sudo nano /lib/systemd/system/rainbot.service
```



The controller takes the output from the two panels and converts it into a digital signal. This is then connected to the Pi, which also supplies 5V to the controller



▲ Make sure everything fits well, is secured and that the jumper cables are not going to pop out

Add in the following text:

```
[Unit]
Description=Rainbot
After=multi-user.target

[Service]
Type=idle
ExecStart=/usr/bin/python3 /home/pi/rainbot/rainbot.py

[Install]
WantedBy=multi-user.target
```

Press **CTRL+X** to save and quit out of nano. Now issue the following commands:

```
sudo chmod 644 /lib/systemd/system/rainbot.service
sudo systemctl enable rainbot.service
sudo systemctl daemon-reload
```

Reboot the Pi. The script will start on reboot (although you won't see any output). Test it with water again.

09 Make it your own

There are lots of improvements that can be made, which we'll leave up to you to explore. Pushover is convenient, but the function could be easily replaced with, well, anything you like. The frequency of checks could be altered (it's currently every five seconds). How about adding an analogue to digital converter and use the A0 output to gauge how heavily it's raining? It's also a great start for a weather station project if you start recording the data. One useful addition would be adding a button for safe shutdown of the Pi after use.

rainbot.py

DOWNLOAD THE FULL CODE: magpi.cc/DAuqUT

Language: **Python 3**

```
001. from gpiozero import DigitalInputDevice
002. from time import sleep
003. import http.client, urllib.parse
004.
005. # Some setup first:
006. APP_TOKEN = 'YOUR_PUSHOVER_APP_TOKEN'    # The app token - required for Pushover
007. USER_TOKEN = 'YOUR_PUSHOVER_USER_TOKEN'   # Ths user token - required for Pushover
008.
009. # Set up our digital input and assume it's not currently raining
010. rainSensor = DigitalInputDevice(17)
011. dryLastCheck = True
012.
013. # Send the pushover alert
014. def pushover(message):
015.     print(message)
016.     conn = http.client.HTTPSConnection("api.pushover.net:443")
017.     conn.request("POST", "/1/messages.json",
018.       urllib.parse.urlencode({
019.         "token": APP_TOKEN,    # Insert app token here
020.         "user": USER_TOKEN,    # Insert user token here
021.         "title": "Rain Detector",
022.         "message": message,
023.       }), { "Content-type": "application/x-www-form-urlencoded" })
024.     conn.getresponse()
025.
026. # Loop forever
027. while True:
028.
029.     # Get the current reading
030.     dryNow = rainSensor.value
031.     print("Sensor says: " + str(dryNow))
032.
033.     if dryLastCheck and not dryNow:
034.
035.         pushover("It's Raining!")
036.
037.     elif not dryLastCheck and dryNow:
038.
039.         pushover("Yay, no more rain!")
040.
041.     # Remember what the reading was for next check
042.     dryLastCheck = dryNow
043.
044.     # Wait a bit
045.     sleep(5)
```

More Minecraft projects with the Wolfram Language

Control an LED with the click of a virtual button

MAKER

Jon McLoone

Jon McLoone holds a degree in Mathematics from the University of Durham and is the Director of Technical Services, Communication and Strategy at Wolfram Research Europe.

jon.mcloone.info

Using the MinecraftLink package available with the Wolfram Language (for setup, see **magpi.cc/Luhhlw**), we'll render a photo in Minecraft by arranging blocks, create replicas of countries by placing blocks at different heights, and position blocks using information from sample CT scan image data – all with a few lines of code.

You'll Need

- Updated Raspbian
- MinecraftLink Library `PacletInstall ["MinecraftLink"]` in Mathematica
- 3D image data from a CT scan (optional)

Rendering a photo

01 Select images

The MinecraftLink package includes MinecraftBlock Entity data from the Wolfram Data Repository. We can use images included in some of those entities to figure out each Minecraft block's average colour. Select all the entities which have images available (except transparent, soft, fire, or water blocks as they don't work well with the game physics):

```
EntityList[Entity["MinecraftBlock",
"Image" -> ImageQ]];
available = Complement[EntityList[Entity
["MinecraftBlock", "Image" -> ImageQ]],
{glass MINECRAFT BLOCK, leaves MINECRAFT BLOCK,
cobweb MINECRAFT BLOCK, sand MINECRAFT BLOCK,
gravel MINECRAFT BLOCK, snow MINECRAFT BLOCK,
fire MINECRAFT BLOCK, water stationary MINECRAFT BLOCK}]
```

Here are the images we have:

```
Magnify[{#["Image"], #} & /@ available,
0.6]
```

02 Picking colour faces

Most blocks (subject to lighting) are the same on all faces, but a few have different textures on their side faces than their top faces. We want to figure out what the blocks' average side-face colour is. To do this, we created the following mask for the position of the side-face pixels of the gold block:

```
mask = Erosion[DominantColors[ , 4,
"CoverageImage"][[2]], 2]
```

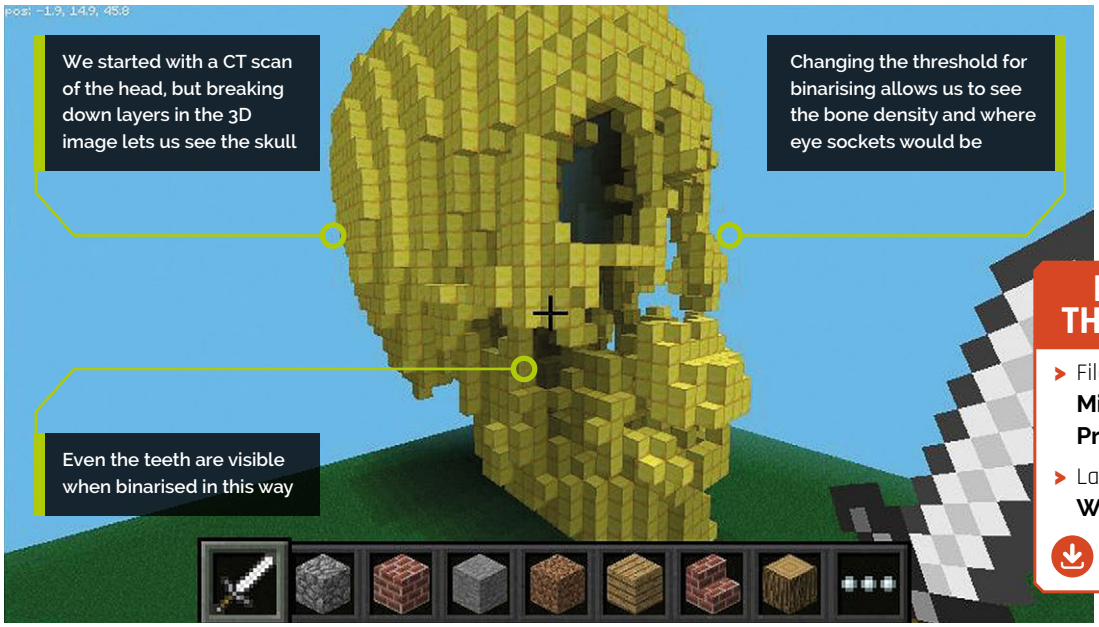
Because all the images have the same shape and viewpoint, we can apply that mask to every block to pick out their front-face pixels:

```
mask RemoveAlphaChannel[Entity[
"MinecraftBlock", "WoodBirch"]["Image"]]
```

03 Removing transparency layer

To make sure we are using a like-for-like measurement, we can remove the transparency layer (using `AlphaChannel`) and put them all into the same colour space. We've asked for the average pixel value and convert that back to an average colour (working in HSB – hue, saturation, brightness – colour gives more perceptually correct averaging of colours):

```
averageColor[block_] :=
Hue[ImageMeasurements[ColorConvert
[RemoveAlphaChannel[block["Image"],
LightBlue], "HSB"], "Mean", Masking →
mask]];
colors = Map[averageColor, available]
```


We started with a CT scan of the head, but breaking down layers in the 3D image lets us see the skull
Changing the threshold for binarising allows us to see the bone density and where eye sockets would be
Even the teeth are visible when binarised in this way

DOWNLOAD THE FULL CODE:

- File name: **Minecraft ProjectCode.nb**
- Language: **Wolfram Language**

magpi.cc/DJegxv

This gives our available palette:

```
ChromaticityPlot[colors]
```

04 Picking out block name

Using `Nearest`, we can take a colour and pick out the block name that is nearest in colour (as the Wolfram Language already knows about perceptual colour distance):

```
getName[color_] :=
First[Nearest[MapThread[Rule, {colors,
available}], color]];
```

Now we need a function that will take a picture and drop its resolution to make it more 'blocky' and simplify the image to use only the colours that are available to us:

```
toBlockColors[img_, size_] := ColorQuantize
[ImageResize[img, size], colors];
```

05 Applying to an image

Let's apply that to a well-known picture:

```
toBlockColors[ , 50]
```

Now we just have to count through the pixels of that image, find the name of the block with the nearest colour to the pixel, and place it in the corresponding place in the Minecraft world:

```
putPicture[{x0_, y0_, z0_}, img_] :=
  Block[{dims = ImageDimensions[img]},
 Do[
  MinecraftSetBlock[{dims[[1]] - x + x0,
y + y0, z0}, getName[RGBColor[ImageValue[img,
{x, y}]]]],
  {x, dims[[1]]}, {y, dims[[2]]}]];
```

06 Running the program

Find a big open space and run the program on a simple image:

```
putPicture[{30, 0, 0}, toBlockColors[
, 50]]
```

07 Importing images

You can use 'Import' to bring images into the system, but fortunately, the Wolfram Language provides lots of images as part of its 'Entity' system. For example, you can fetch some very famous works of art:

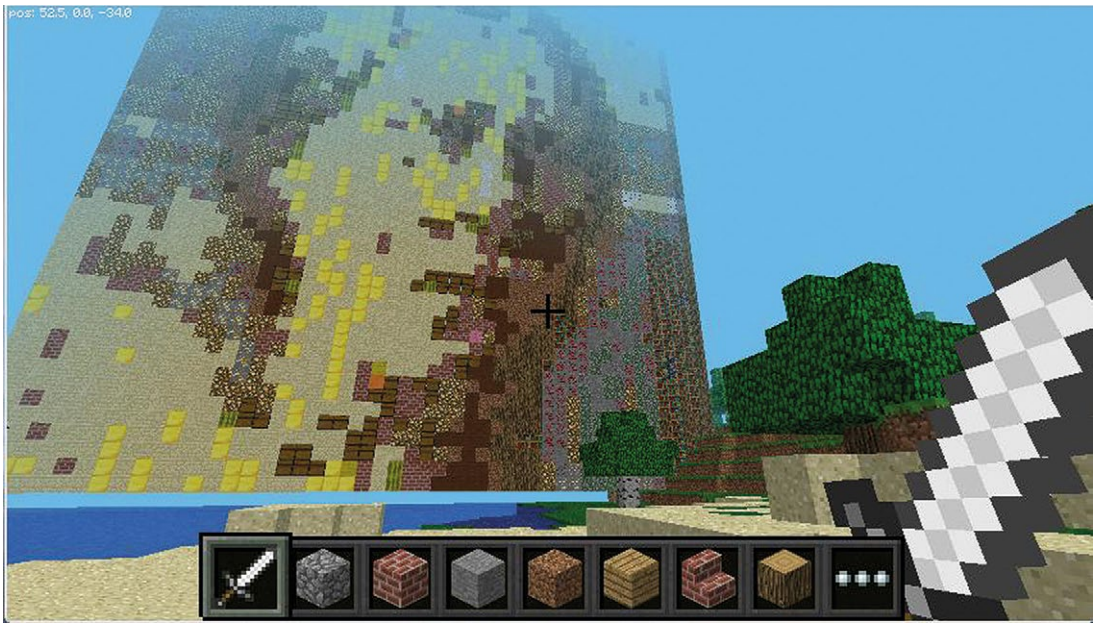
```
American Gothic ARTWORK ["Image"]
```

Here is a detail from *American Gothic* (Grant Wood's sister) in blocks:

Get The MagPi 72

This is the latest installment in a series of Minecraft and Mathamatica tutorials. You can download digital editions of previous tutorials for free. Start with *The MagPi #72*

magpi.cc/72



Importing famous works of art is easy using the Wolfram Language, and once you have one you can project it into your Minecraft world

```
putPicture[{30, 0, 0},
toBlockColors[ImageTake[ American Gothic ARTWORK
["Image"], {25, 75}, {10, 50}], 50]]
```

Recreating the real world

08 Getting data

Let's suppose we want to create the United Kingdom in Minecraft. All we need to do is place a grid of blocks at heights that correspond to heights of the land in the UK. We can get that data from the Wolfram Language with `GeoElevationData`:

```
ListPlot3D[Reverse /@ GeoElevationData[
United Kingdom COUNTRY ], PlotRange → {-10 000,
20 000}, Mesh → False]
```

You will see that the data includes underwater values, so we will need to handle those differently to make the shape recognisable. Also, we don't need anywhere near as much resolution (`GeoElevationData` can go to a resolution of a few metres in some places). We need something more like this:

```
ListPlot3D[Reverse /@ GeoElevationData[
United Kingdom COUNTRY , GeoZoomLevel → 3],
PlotRange → {0, 5000}, Mesh → False]
```

09 Creating blocks

Now let's make that into blocks. Let's assume we will choose the minimum and maximum heights of our output. For any given position, we need to create a column of blocks. If the height is positive, this should be solid blocks up to the height, and air above. If the height is negative, then it is solid up to the point, water above that until we reach a given sea level value, and then air above that.

```
createMapColumn[{x_, y_, z_}, seaLevel_,
min_, max_] := MinecraftSetBlock[{{x, min,
z}, {x, y, z}}, "Dirt"];
    If[y ≥ seaLevel, MinecraftSetBlock[{{x,
y, z}, {x, max, z}}, air MINECRAFTBLOCK ],
     MinecraftSetBlock]
{{x, y, z}, {x, seaLevel - 1, z}},
water stationary MINECRAFT BLOCK ];
     MinecraftSetBlock[{{x, seaLevel, z},
{x, max, z}}, "Air"]] ;
```

10 Creating columns

Now we just need to create a column for each position in our elevation data.

All the work is in transforming the numbers. The reversing and transposing is to get the co-ordinates to line up with the compass correctly; `QuantityMagnitude` gets rid of units, and the rest is vertical scaling:

```
 MinecraftElevationPlot[data0_, {x0_,
seaLevel_, z0_}, maxHeight_: 5] :=
  Block[{data =
QuantityMagnitude[Reverse[Map[Reverse,
Transpose[data0]]]],
 scale, min, dims},
  dims = Dimensions[data];
  scale = maxHeight / Max[Flatten[data]];
  min = Round[scale * Min[Flatten[data]]];
  Do[createMapColumn[{Round[x0 + i],
Floor[scale data[[i, j]] + seaLevel],
z0 + j}, Round[seaLevel], seaLevel + min,
Round[maxHeight + seaLevel]], {i, dims[[1]]},
{j, dims[[2]]}]]
```

11 Clearing a flat area

Before we start, we can use the following code to clear a large, flat area to work on and put the camera high in the air above the action:

```
 MinecraftSetBlock[{{-40, -10, -40}, {40, 0,
40}}, "Grass"];
 MinecraftSetBlock[{{-40, 0, -40}, {40, 50,
40}}, "Air"];
 MinecraftSetCamera["Fixed"];
 MinecraftSetCamera[{0, 25, 0}]
```

12 Placing the map

And now we can place the map:

```
 MinecraftElevationPlot[
GeoElevationData[ United Kingdom COUNTRY ,
GeoZoomLevel → 2], {-15, 0, -15}, 5]
```

You can just see that the land is higher in mountainous Scotland. You can see that better with the camera in the usual position, but the coastline becomes harder to see.

13 Creating Mount Everest

Alternatively, here is the view of the north ridge of Mount Everest, as seen from the summit:

```
 MinecraftSetCamera["Normal"]
 MinecraftElevationPlot[GeoElevationData[
GeoDisk[ Mount Everest MOUNTAIN , 3 mi],
GeoZoomLevel → 9], {-15, -18, -15}, 30]
```

A nicer version of this might switch materials at different heights to give you snowcapped mountains, or sandy beaches. Try them out for yourself!

Rendering a CT scan

14 Getting built-in example

You can use this built-in example if you don't have your own CT scan image data available:

```
 Show[ExampleData[{"TestImage3D", "CThead"}],
BoxRatios → 1]
```

Our simplest approach is just to drop the scan's resolution, and convert it into either air or blocks:

```
 Binarize[ImageResize[ExampleData[
{"TestImage3D", "CThead"}], {80, 80, 80}]]
```

15 Finding solid voxel co-ordinates

We can easily find the co-ordinates of all the solid voxels (3D pixels), which we can use to place blocks in our world:

```
Position[ImageData[%], 1]
```

16 Wrapping into a single function

We can wrap all of that into a single function, and add in an initial position in the Minecraft world. We added the small pause because if you run this code from a desktop computer, it will flood the Minecraft server with

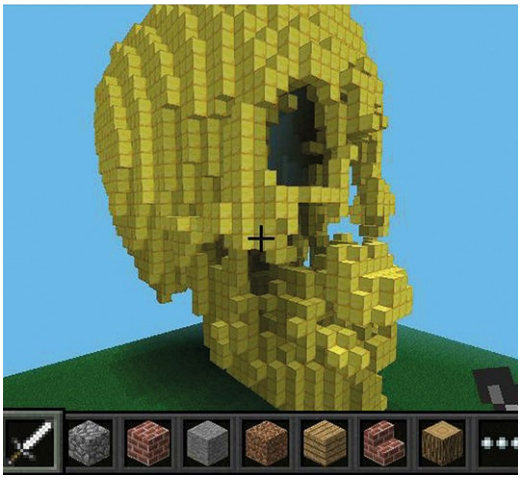
more requests than it can handle, and Minecraft will drop some of the blocks:

```
 fixCoordinates[{a_, b_, c_}] := {c, -a, b}
(*Mapping coordinate systems*)
```

```
 minecraftImage3D[img_Image3D, pos_List,
block_, threshold_: Automatic] := (
   MinecraftSetBlock[{pos, pos +
ImageDimensions[img]}, "Air"];
   Map[(Pause[0.01];
      MinecraftSetBlock[pos + #, block]) &,
fixCoordinates /@
Position[ImageData[Binarize[img, threshold]],
1]];)
```

And here it is in action with the head:

```
 minecraftImage3D[
  ImageResize[ExampleData[{"TestImage3D",
"CThead"}], {40, 40, 40}], {0, 40, 0},
"GoldBlock"]
```



17 Picking out internal layers

But one thing to understand with 3D images is that there is information 'inside' the image at every level, so if we change the threshold for binarising then we can pick out just the denser bone material and make a skull:

```
 Binarize[ImageResize[ExampleData
[{"TestImage3D", "CThead"}], {80, 80, 80}],
0.4]
```

```
 minecraftImage3D[
  ImageResize[ExampleData[{"TestImage3D",
"CThead"}], {40, 40, 40}], {0, 40, 0},
"GoldBlock", 0.4]
```

An interesting extension would be to establish three levels of density and use the glass block type to put a transparent skin on the skull. We'll leave that for you to do. You can find DICOM medical images on the web that can be imported into the Wolfram Language with 'Import', but beware – some of those can be quite large files.

Automatic pyramid building

18 Scanning surface blocks

Finally, we can create new game behaviour with a special block combination that triggers an automatic action – when you place a gold block on top of a glowstone block, a large pyramid will be built for you.

The first step is to scan the surface blocks around a specific point for gold and return a list of surface gold block positions found:

```
 scanForGold[{x0_, y0_, z0_}]
:= Block[{goldPos = {}, height =
MinecraftGetHeight[{x, z}]},
    Do[Pause[0.1];
 If[MinecraftGetBlock[{x, height - 1, z}]
=== Entity["MinecraftBlock", "GoldBlock"],
AppendTo[goldPos, {x, height - 1, z}]],
 {x, x0 - 1, x0 + 1}, {z, z0 - 1, z0 + 1}];
    goldPos];
```

19 Checking for glowstone

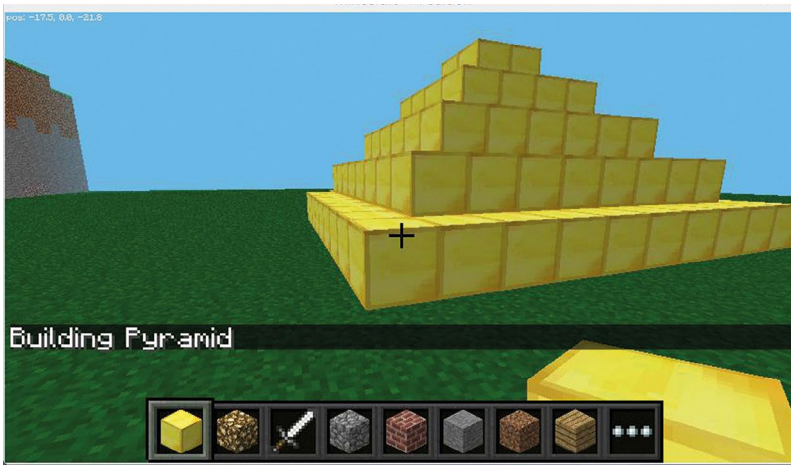
Next, look under each of the gold blocks that we found and see if any have glowstone under them:

```
 checkGoldForGlowstone[goldPos_] :=
FirstCase[goldPos, {x_, y_, z_} /;
MinecraftGetBlock[{x, y - 1, z}] ===
Entity["MinecraftBlock", "GlowstoneBlock"]]
```

20 Setting up the pyramid

Now we need a function that performs the resulting actions. It posts a message, removes the two special blocks and sets the pyramid:

```
 pyramidActions[found_] :=
(MinecraftChat["Building Pyramid"];
    MinecraftSetBlock[{found, found - {0, 1,
0}}, "Air"];
    MinecraftSetBlock[found - {0, 1, 0},
"GoldBlock", Pyramid[], RasterSize → 12]);
```



21 Matching location

We can now put all of that together into one function that scans around the current player and runs the actions on the first matching location. The `PreemptProtect` is a bit subtle. Because we are going to run this as a background task, we need to make sure that we don't perform two actions at once, as the messages going back and forth to the Minecraft server may get muddled:

```
 pyramidCheck[] := PreemptProtect
[Block[{found = checkGoldForGlowstone
[scanForGold[MinecraftGetTile[]]]},
If[Not[MissingQ[found]],
pyramidActions[found]]]]
```

All that's left is to run this code repeatedly every five seconds:

```
 task = SessionSubmit
[ScheduledTask[pyramidCheck[], 5]]
```

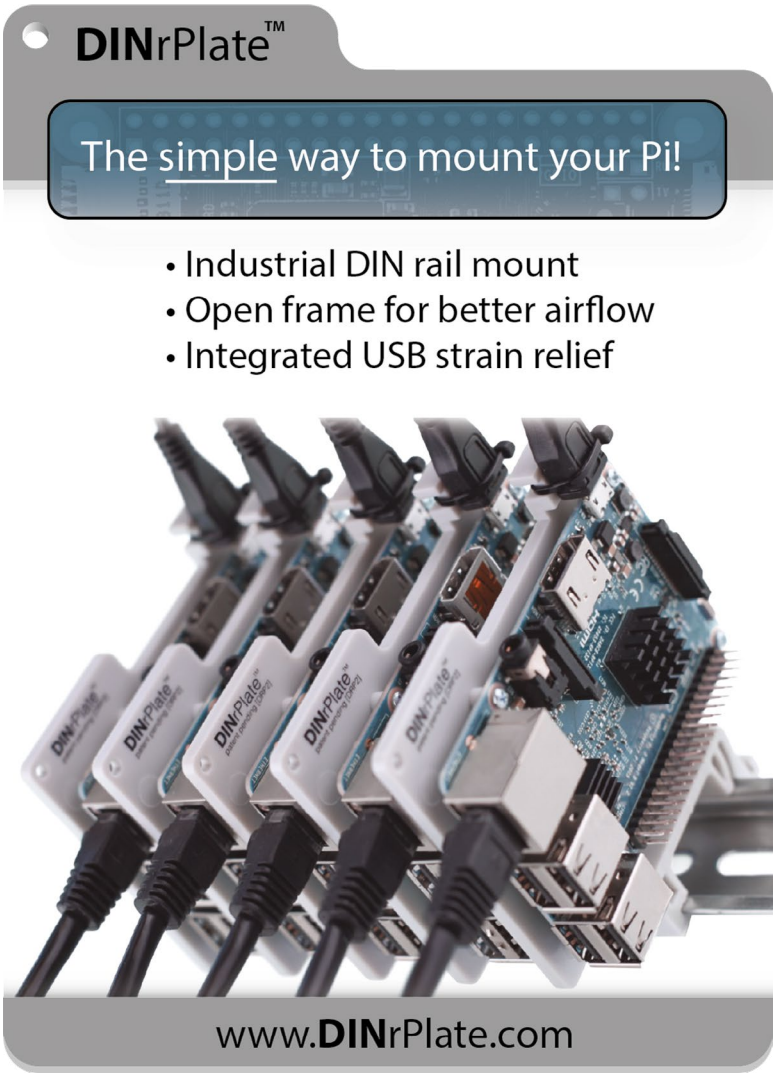
22 Placing blocks

We place the blocks and walk up within one block of the special column and wait for a couple of seconds, until the game chat reads 'Building Pyramid' and the pyramid appears in the game.

23 Stopping the task

To stop the task, you can evaluate:

```
TaskRemove[task]
```


DINrPlate™
The simple way to mount your Pi!
• Industrial DIN rail mount
• Open frame for better airflow
• Integrated USB strain relief
www.DINrPlate.com

DOWNLOAD THE FULL CODE:
magpi.cc/ztqwni

Part 10

Coding games on the Raspberry Pi in C/C++

Let's go retro and make our game look better

MAKER
Brian Beuken

Very old game programmer now teaching very young game programmers a lot of bad habits at Breda University of Applied Science in Breda NL.

magpi.cc/YxaUVQ

You'll Need

- Code::Blocks `sudo apt-get codeblocks`
- FreeType2
- stb_image.h (see last issue)

Maths can be the most daunting part of any new coder's journey. So far we've managed to avoid doing too much, but there has been no way to escape it: we had to scale things, reposition things, and also to use concepts like vectors. Despite keeping the maths very simple, we have had to do rather a lot of it. And though it might not seem like it, it's actually quite a bit harder to write code without the standard maths, and it made us jump through a few hoops as we've progressed.

Maths is fun...

No, really it is, but we need to realise the benefits we get from it. We've been using `SCALEFACTOR` to multiply and divide screen positions, and let us work out where things are. That is all fine, but we did have to do a lot of it, didn't we? And we also ended up with a lot of `(((())))` brackets to keep the code in place. Also, though we might not notice it yet, our shaders were using such hacky slow maths, it makes old coders cry when they look at it. We've actually been making our life difficult by trying so hard to use only the simplest maths, so it's time for us to take a step into the light and begin to embrace some more complex mathematical ideas.

C++ comes with a standard set of maths functions, but they don't provide the kind of systems we normally need in 2D and 3D graphics, like vectors and matrices. Lucky for us, though, we don't need to write, or even to fully understand (at first) all the different kinds of useful features that are available to us. There are several different maths libraries available to download. We're going to use one of the most popular for game programming that makes use of OpenGL; it's called GLM or OpenGL Mathematics.

GLM rocks

GLM is popular due to its ability to be used across different systems, and its adherence to the maths system commonly used in our OpenGL Shader language, so you'll find it in Raspbian, Windows, UNIX, and almost every other type of OS that can compile its own code and makes use of OpenGL.

Because it's a source code library, we do have to download and save it in a directory – not the **tmp** dir on our Raspberry Pi, since that will be cleared every time we power the system down. Raspbian is a little protective of its directories, so usually it's best to create a new **GLM** dir within the **/home/pi** dir, like we did before with STB. We will assume in this lesson's code that GLM is located at **/home/pi/GLM**. GLM updates from time to time, so it's best to do a web search for 'GLM maths': the current version can be downloaded from **glm.g-truc.net**. Once downloaded and unzipped to a directory, all we need to do is add the **GLM** dir to our compiler `include` directories, and we can add `<glm.hpp>` when we need it.

What we have now is so much better than SimpleVec2 with its basic container functionality. We have a full set of vec2, vec3, even vec4 functions, as well as a few other special types of data, along with a range of different kinds

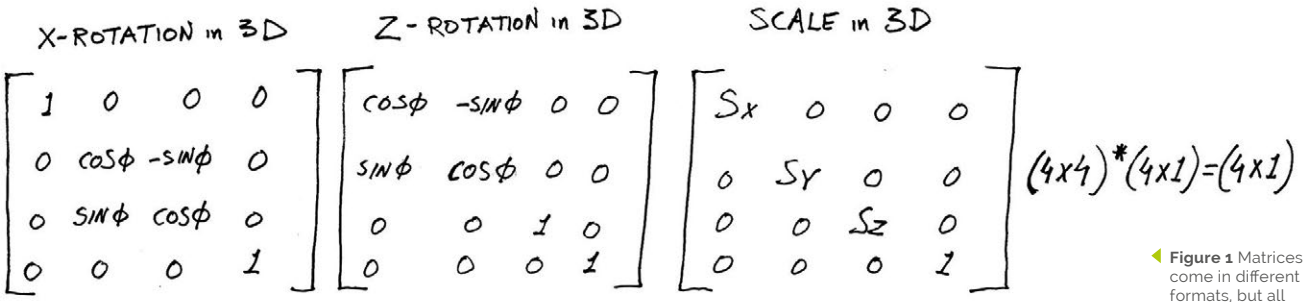
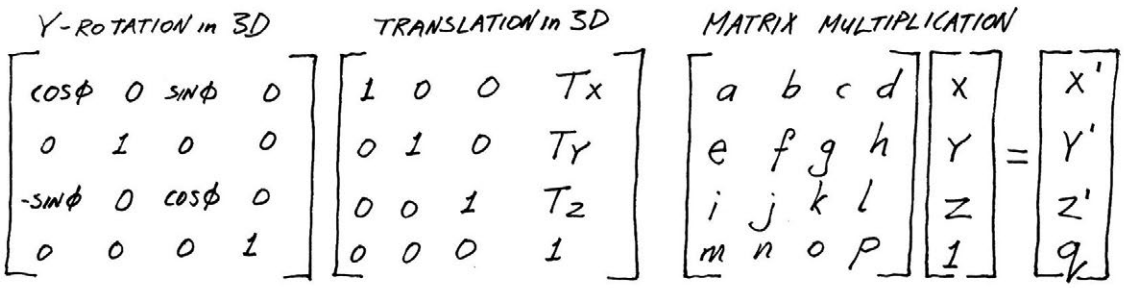


Figure 1 Matrices come in different formats, but all 'transform' a vector



of matrices. But, also importantly, we have the means to let them interact so we can multiply vertices, matrices, and even vectors by matrices.

But wait, we're getting ahead of ourselves: why would we want vectors and matrices to interact, and what exactly are these things anyway?

Vectors represent a few different things on computers. They can be used to indicate directions, forces, intensity, as well as a general grouping of data in twos, threes and fours, known as tuples. For example, we can represent a point in space, as x and y co-ordinates, and we can represent a direction as x steps along, y steps up. In 3D we can also add a z axis to get directions or positions anywhere in 3D space.

What a transformation

So, vectors are flexible, especially for our main requirement as positions. However, we know that if we want to manipulate things to change their size or position, we need to do some maths. Moving, also known as translating – where we move a point from one place in space to another – we understand; scaling is fairly easy to wrap our heads around, too.

In computer terms, we refer to scaling and translating as types of transformation. The transformation results in a point being at a different location in space. We use multiple points to define our tiles and sprites, and indeed many thousands when we define a complex 3D shape. So a lot of transformations have to be applied to each point, or *vertex* as it is properly known.

There are 'usually' three types of transformation possible. The third we haven't looked at yet is rotation, where we can rotate a point around an origin – **see Figure 3** (overleaf).

When you consider the maths needed to do scaling, translation, and now rotation all together, you discover you have in fact a lot of individual calculations to do to 'transform' a point or vertex from one place to another. It's just about manageable in 2D, as we have seen, but when we branch out to 3D the number of calculations increases vastly.

> We know that if we want to manipulate things to change their size or position, we need to do some maths

And that is where matrices come in, although they need a bit of setting up: we have to plug data into the right part of the matrix. Once set up, they are capable of doing all transformations seemingly with one function (in reality there's

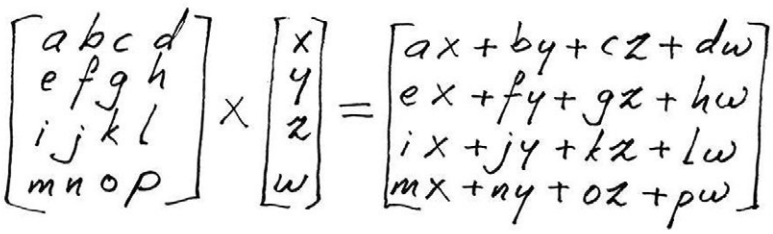


Figure 2 A matrix does all this work with a simple V*M

Top Tip

Have a window option

Even though we now have a full-screen display, being able to switch back to window makes placement and observation easy.

Top Tip

Take note of assets

We now have an asset manager to take care of things for us; be careful not to bypass it.

still a lot of maths happening, but at a level below where we are watching).

Computers really prefer to use data in vectors and matrix arrangements, which allows them to be used over and over again, so a small penalty in the setup is rewarded manyfold in the reuse of data to transform all the vertices a standard game needs to convert.

Matrices are funny things, though. Each transform has its own layout of a matrix, so scale, translation, and rotation are all done individually. In fact, rotation has three versions: one for each axis, xyz. For 2D we only really need the rotation around the z axis to create spinning.

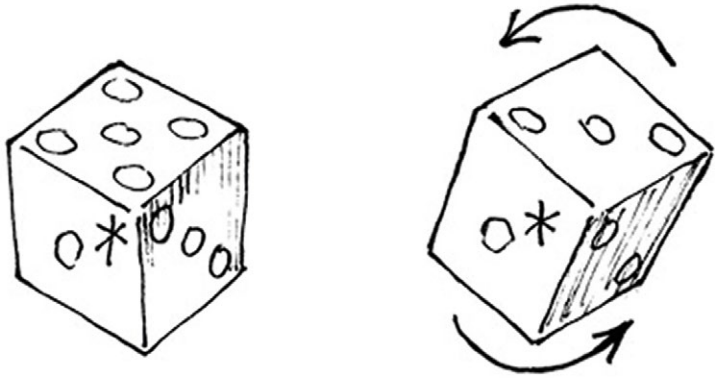
> "In games, death is usually not the end. The challenge we have is in detecting when our character needs to die"

Once all the matrices are set up, we can combine them to create a master transform matrix, which can be used on every single point we need to change. In **Figure 1** and **2**, we've shown work on a vec4.

Our Lesson 10 demo has examples of working with the sprites using matrices and vectors; they now have an additional `Rotation` function added. Our `Sprites` shader is now using matrices and vectors and though we won't really notice it on such a small project, we have dramatically improved the performance of that shader.

Tiles don't really need to spin, so we'll leave them as they are with their fast VBO setup, but feel free to change them if you want.

Figure 3 Objects need to have an origin to rotate



Game states

The concept of states was introduced with our Wake and Chase baddies, but our games – like our characters – also have states, and it's common to find a game project broken down into such states as Menu, Game, HighScore, etc. This also allows us to keep track of and maintain separate classes or code areas to handle such things.

Our new demo has a menu class and, as you can see, it uses the font drawing code we discovered last time to allow us to take input and set a few game parameters.

Once we're ready to start the game, we can transition our state to a gameplay mode, set it up ready for a new player, and let it cycle and reset, until all the levels are done, before producing an end game state and going back to our start menu.

Death and success

In games, a death – by being caught/stomped on, drowned, or otherwise prevented from living – is usually not the end. We have the lives concept where we lose a life, perhaps set our player back to the start, and keep track of whether our lives have run out before heading back to the menu. The only small challenge we have is in detecting when our character needs to die, and by what means. For the most part we'll drown him if he's in water for more than a few seconds, and if any enemy touches Bob, we'll give him a 'fall off the map' state, leading to a death condition which lets us test if the game is over. The source code should make this very clear, so check it out.

Success is a rewarded, usually by the setup and playing of a new level, maybe with a few bonuses such as an extra life, which we can now display with our font display system.

Not everything we touch is gold

Being touched by one of our enemies as a kill condition is a pretty traditional game mechanic, but we can also touch and collect other objects which give us more lives or enhance our abilities, or indeed change our state of being.

We generally refer to such things as 'pick-ups', which, like any other object in our game, have a small piece of logic to detect a collision – and, if one is found to occur, to act accordingly, increasing the life counter or adding points to our score.

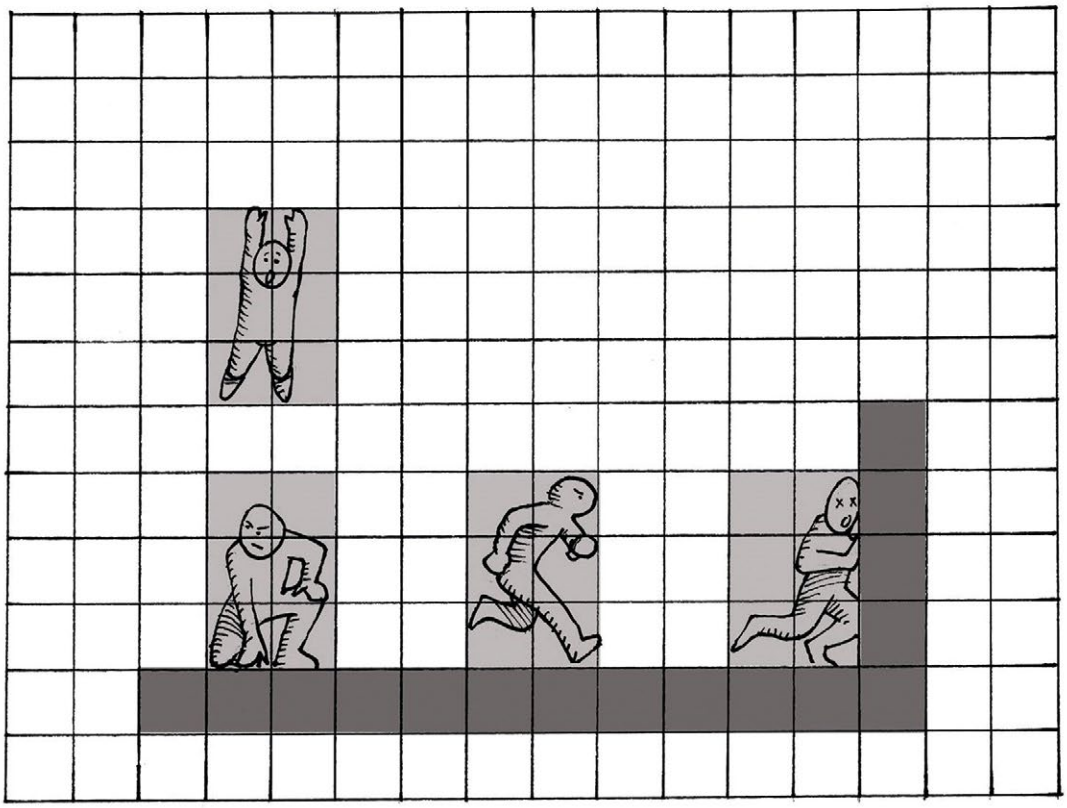


Figure 4 Bumping into walls is not much different from landing

Let's keep track of things

Finally this issue, let's take note of something quite important: our assets! Have you noticed that even though we are using the same graphics for our fungus-type baddies, we reload the graphics for every single instance? If we have five baddies on screen, we're loading the graphics five times. That honestly isn't needed and it's also quite a serious issue for a machine with fairly limited memory like our Raspberry Pi.

We can overcome this easily by creating a small utility class to keep track of the assets we load, and if already loaded, to return the location in memory of the assets from the first time it was loaded. The new `AssetManager` class takes care of this. It has a `LoadAsset` method which will keep an internal note of all files we load, and store them in a special type of data structure called a *map*. This is a bit like a 2D array, but instead of indexes, it has the names of the assets associated with the location or information structure we created when they first loaded. That enables multiple instances to use the same data, either in its raw file format or, more effectively, in its converted to GPU texture / VBO format.

Review the source code to see how that works. Maps are incredibly useful for jobs like this, where you want to keep track of things by name.

One more thing to add is some better maps. This version of the game now has four maps and we've made our ability interact with the maps more variable by using pointers to maps. That allows for considerably more expansion as we slowly remove our dependence on hard numbers.

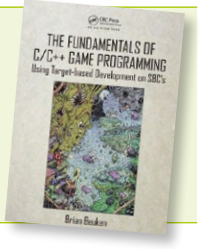
Next time

Our game is pretty much complete, but there are rough edges. For example, we can walk through walls and steps. Try adding checks to the direction of Bob's movement to prevent that, the same way we stop him falling – see **Figure 4**.

Next time we'll add a few FX to our shaders and jazz things up a bit, as well as fixing some of our timing issues.

Learn more about C

Brian has a whole book on the subject of making games C and C++, called *The Fundamentals of C/C++ Game Programming: Using Target-based Development on SBCs*. Grab it here: **magpi.cc/nUkjEt**



Top Tip

Avoid magic numbers

We still have too many hard numbers. Try to go through the code and see how you can replace them.

Raspberry Pi QuickStart Guide

Setting up your Raspberry Pi is pretty straightforward. Just follow the advice of **Rosie Hattersley**

Congratulations on becoming a Raspberry Pi explorer. We're sure you'll enjoy discovering a whole new world of computing and the chance to handcraft your own games, control your own robots and machines, and share your experiences with other Pi fanatics.

Getting started won't take long: just corral all the bits and bobs on our checklist, plus perhaps a funky Pi case to house it. Useful extras include some headphones or speakers if you're keen on using your Raspberry Pi as a media centre or gaming machine.

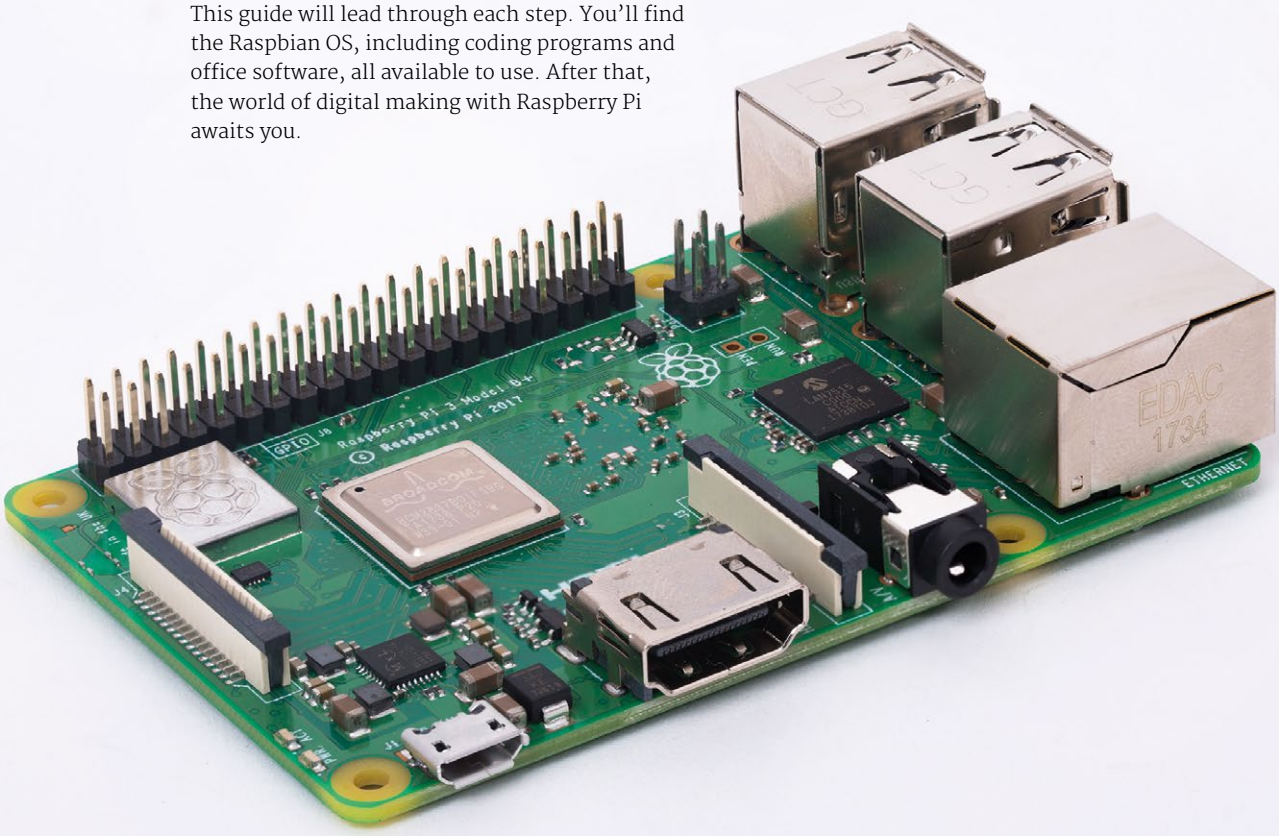
To get set up, simply format your microSD card, download NOOBS, and run the Raspbian installer. This guide will lead through each step. You'll find the Raspbian OS, including coding programs and office software, all available to use. After that, the world of digital making with Raspberry Pi awaits you.

What you need

All the bits and bobs you need to set up a Raspberry Pi computer

A Raspberry Pi

Whether you choose a Raspberry Pi 3B+, 3B, Pi Zero, Zero W, or Zero WH (or an older model of Raspberry Pi), basic setup is the same. All Raspberry Pi computers run from a microSD card, use a micro USB power supply, and feature the same operating systems, programs, and games.

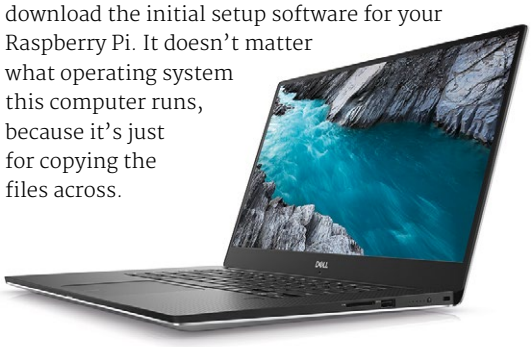


8GB microSD card

You'll need a microSD card with a capacity of 8GB or greater. Your Raspberry Pi uses it to store games, programs and photo files and boots from your operating system, which runs from it. You'll also need a microSD card reader to connect the card to a PC, Mac, or Linux computer.

Mac or PC computer

You'll need a Windows or Linux PC, or an Apple Mac computer to format the microSD card and download the initial setup software for your Raspberry Pi. It doesn't matter what operating system this computer runs, because it's just for copying the files across.



USB keyboard

Like any computer, you need a means to enter web addresses, type commands, and otherwise control your Raspberry Pi. You can use a Bluetooth keyboard, but the initial setup process is much easier with a wired USB keyboard.

USB mouse

A tethered mouse that physically attaches to your Raspberry Pi via a USB port is simplest and, unlike a Bluetooth version, is less likely to get lost just when you need it. Like the keyboard, we think it's best to perform the setup with a wired mouse.

Power supply

The Raspberry Pi uses the same type of USB power as your average smartphone. So you can recycle an old USB to micro USB cable and smartphone power supply. Raspberry Pi also sells an official power supply (**magpi.cc/universalpower**), which provides a reliable and steady source of power.

Display

A common-or-garden PC monitor is ideal, as the screen will be large enough to read comfortably. It needs to have an HDMI connection, as that's what's fitted on the Raspberry Pi board.

USB hub (for Pi Zero W)

The Pi Zero W doesn't have standard-size USB ports. Instead it has a micro USB port (and usually comes with a micro USB to USB adapter). To attach a keyboard and mouse (and other items) to a Pi Zero, you should get a four-port USB hub (or use a keyboard with a hub built in).

Set up a Raspberry Pi

The Raspberry Pi 3 / 3B+ has plenty of connections, making it easy to set up

01 Hook up the keyboard

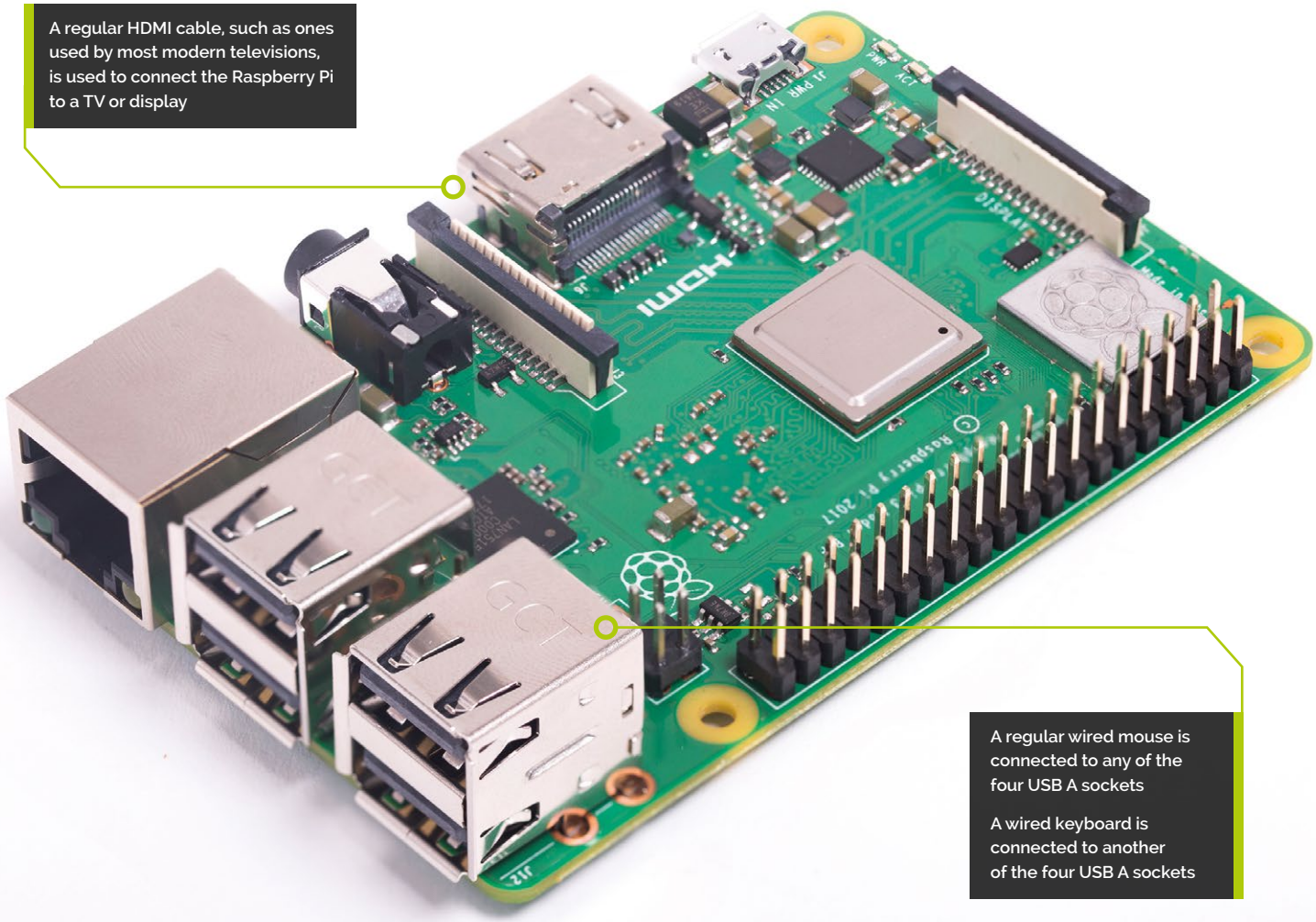
Connect a regular wired PC (or Mac) keyboard to one of the four larger USB A sockets on a Raspberry Pi 3/3B+. It doesn't matter which USB A socket you connect it to. It is possible to connect a Bluetooth keyboard, but it's much better to use a wired keyboard to start with.

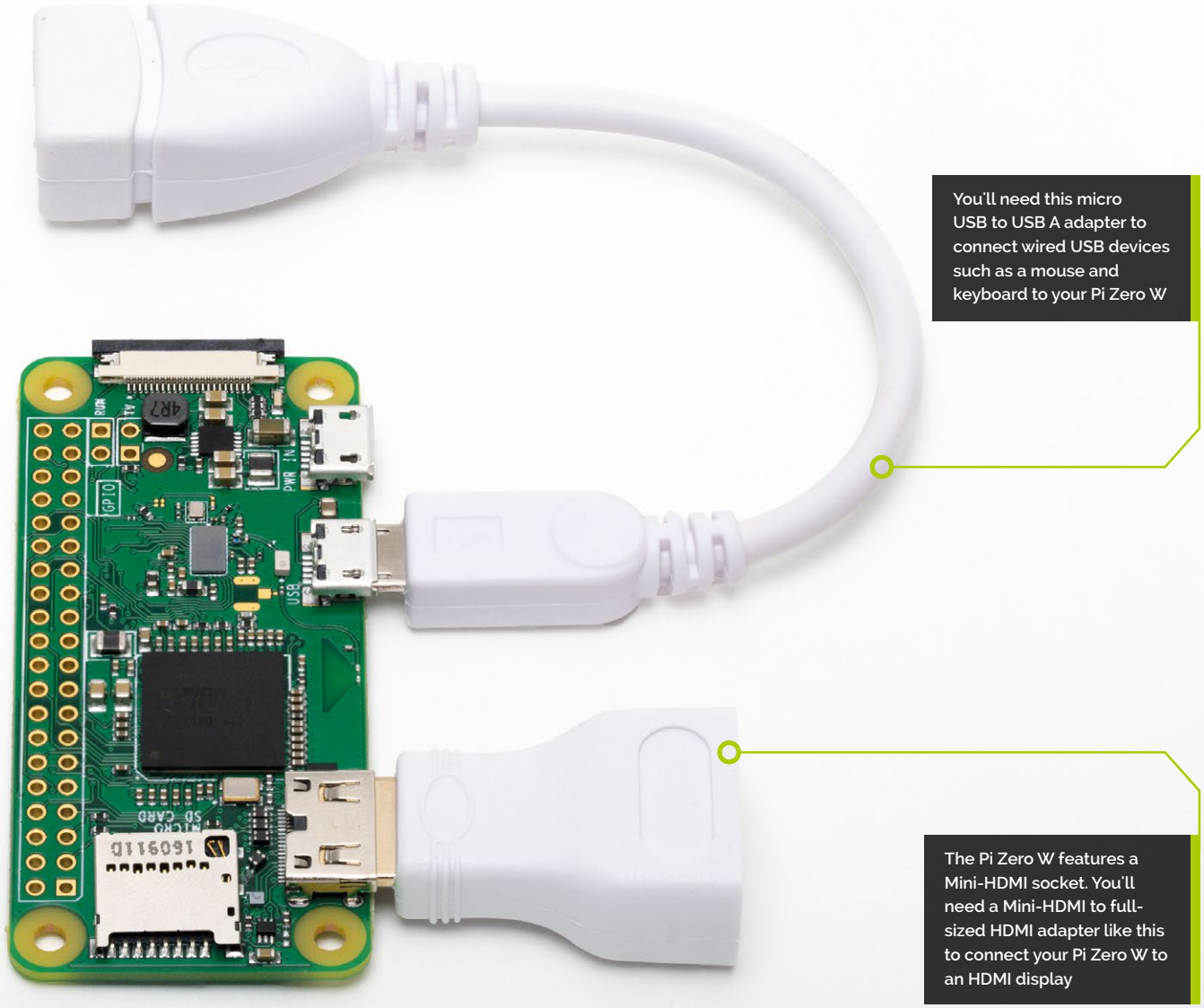
02 Connect a mouse

Connect a USB wired mouse to one of the other larger USB A sockets on the Raspberry Pi. As with the keyboard, it is possible to use a Bluetooth wireless mouse, but setup is much easier with a wired connection.

03 HDMI cable

Next, connect the Raspberry Pi to your display using a full-size HDMI cable. This will connect directly to the HDMI socket on the side of a larger Raspberry Pi 3/3B+. Connect the other end of the HDMI cable to an HDMI monitor or television.


A regular HDMI cable, such as ones used by most modern televisions, is used to connect the Raspberry Pi to a TV or display
A regular wired mouse is connected to any of the four USB A sockets
A wired keyboard is connected to another of the four USB A sockets



You'll need this micro USB to USB A adapter to connect wired USB devices such as a mouse and keyboard to your Pi Zero W

The Pi Zero W features a Mini-HDMI socket. You'll need a Mini-HDMI to full-sized HDMI adapter like this to connect your Pi Zero W to an HDMI display

Set up a Pi Zero

You'll need a couple of adapters to set up a Raspberry Pi Zero / W / WH

01 Get a Pi Zero W connected

If you're setting up a smaller Raspberry Pi Zero, you'll need to use a micro USB to USB A adapter cable to connect the keyboard to the smaller connection on a Pi Zero W. A Pi Zero W has only a single micro USB port for connecting devices, which makes connecting both a mouse and keyboard slightly trickier than when using a larger Raspberry Pi.

02 Mouse and keyboard

You can either connect your mouse to a USB socket on your keyboard (if one is available), then connect the keyboard to the micro USB socket (via the micro USB to USB A adapter). Or, you can attach a USB hub to the micro USB to USB A adapter.

03 More connections

Now connect your full-sized HDMI cable to the Mini-HDMI to HDMI adapter and plug the adapter into the Mini-HDMI port in the middle of the Pi Zero W. Connect the other end of the HDMI cable to an HDMI monitor or television.

Set up the software

Use NOOBS to install Raspbian OS on your microSD card and start your Raspberry Pi

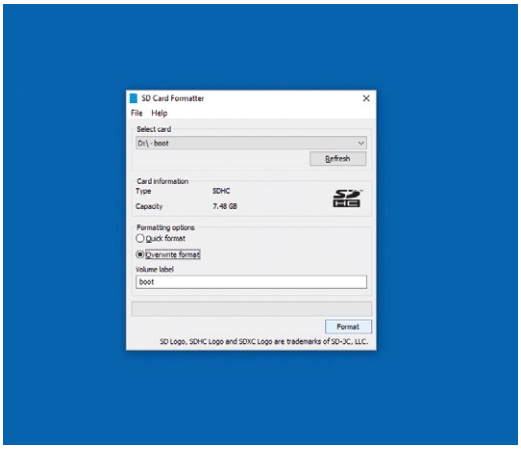
Now you've got all the pieces together, it's time to install an operating system on your Raspberry Pi, so you can start using it.

Raspbian is the official OS for the Raspberry Pi, and the easiest way to set up Raspbian on your Raspberry Pi is to use NOOBS (New Out Of Box Software).

If you bought a NOOBS pre-installed 16GB microSD card (**magpi.cc/huLdtN**), you can skip Steps 1 to 3. Otherwise you'll need to format a microSD card and copy the NOOBS software to it.

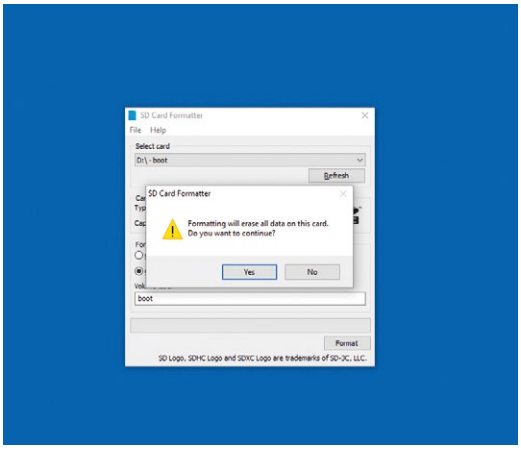
You'll Need

- A Windows/Linux PC or Apple Mac computer
- A microSD card (8GB or larger)
- A microSD to USB adapter (or a microSD to SD adapter and SD card slot on your computer)
- SD Memory Card Formatter **rpf.io/sdcard**
- NOOBS **rpf.io/downloads**



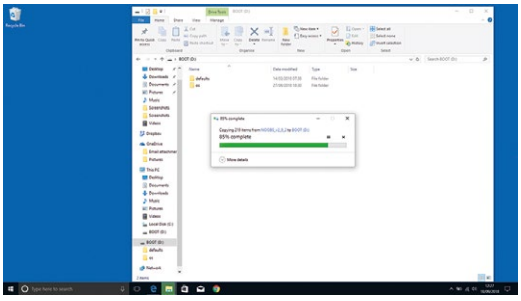
01 Prepare to format

Start by downloading SD Card Formatter tool from the SD Card Association website (**rpf.io/sdcard**). Now attach the microSD card to your PC or Mac computer and launch SD Card Formatter (click Yes to allow Windows to run it). If the card isn't automatically recognised, remove and reattach it and click Refresh. The card should be selected automatically (or choose the right one from the list).



02 Format the microSD

Choose the Quick Format option and then click Format (if using a Mac, you'll need to enter your admin password at this point). When the card has completed the formatting process, it's ready for use in your Raspberry Pi. Leave the microSD card in your computer for now and simply note the location of your duly formatted SD card. Windows will often assign it a hard drive letter, such as E; on a Mac it will appear in the Devices part of a Finder window.



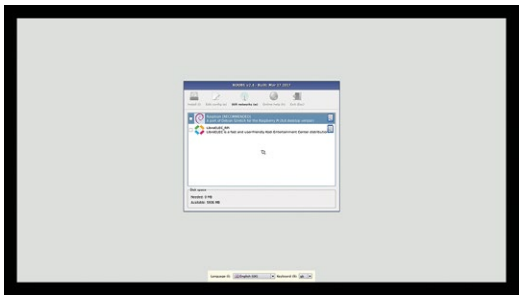
03 Download NOOBS

Download the NOOBS software from **rpf.io/downloads**. NOOBS (New Out Of Box System) provides a choice of Raspberry Pi operating systems and installs them for you. Click 'Download zip' and save the file to your Downloads folder. When the zip file download is complete, double-click to launch and uncompress the folder. You'll need to copy all the files from the NOOBS folder to your SD card. Press **CTRL+A** (**⌘+A** on a Mac) to select all the files, then drag all the files to the SD card folder. Once they've copied across, eject your SD card. Be careful to copy the *files inside* the NOOBS folder to the microSD card (not the NOOBS folder itself).

First, insert your microSD card into the Raspberry Pi
With the microSD card fully inserted, connect your micro USB power cable to the Raspberry Pi. A red light will appear on the board to indicate the presence of power

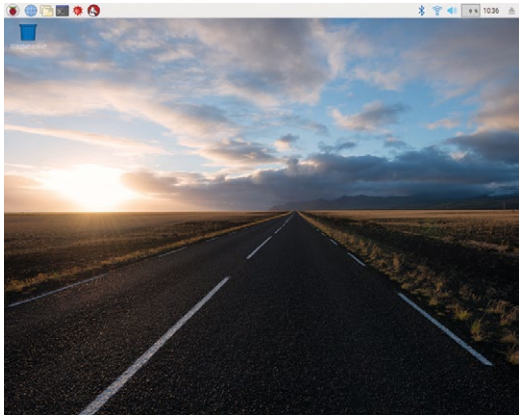
04 Assemble your Raspberry Pi

Now it's time to physically set up your Raspberry Pi. Plug your PC monitor into the mains and attach its HDMI cable to the corresponding HDMI port on your Raspberry Pi. Plug in the power supply and attach the micro USB end to the Raspberry Pi. Use the remaining USB ports to attach keyboard and mouse. Finally, remove the microSD card from the SD card adapter and slot it into the underside of your Raspberry Pi 3B+ or 3. Pi Zero W owners will need to attach a USB hub to connect mouse, keyboard, and monitor. The Zero W's microSD card slot is on the top of its circuit board.



05 Power up

Plug in your Raspberry Pi power supply and, after a few seconds, the screen should come on. When the NOOBS installer appears, you'll see a choice of operating systems. We're going to install Raspbian, the first and most popular one. Tick this option and click Install, then click Yes to confirm. For more OS options, instead click 'Wifi networks' and enter your wireless password; more OS choices will appear. Installation takes its time but will complete – eventually. After this, a message confirming the success installation appears. Your Raspberry Pi will prompt you to click OK, after which it will reboot and load the Raspbian OS.



06 Get online

When Raspbian loads for the first time, you need to set a few preferences. Click Next, when prompted, then select your time zone and preferred language and create a login password. You're now ready to get online. Choose your WiFi network and type any required password. Once connected, click Next to allow Raspbian to check for any OS updates. When it's done so, it may ask to reboot so the updates can be applied.

Click the Raspberry icon at the top left of the screen to access familiar items such as the LibreOffice suite, internet, games, and accessories such as the image viewer, text editor, and calculator. You're all set to start enjoying your very own Raspberry Pi.

FAQ

Your technical hardware & software problems solved...

Magic Mirror

What is a Magic Mirror?

In tech terms

In this instance we don't mean one that will tell you who the fairest of them all is (you don't need magic to know it's our Features Ed, Rob), but instead a Raspberry Pi-powered mirror that displays useful information on its surface as if by magic.

How it works

Traditionally, Raspberry Pi magic mirrors are large screens (usually an old TV) mounted in a frame with a mirror-film coating over it. The screen displays a black image, which is what makes it work as a mirror, and then displays selected info over it. These bits of data can then be seen on the surface.

Info on display

Usually, magic mirror software will let you display the time, local weather, your calendar appointments for the day, and other such useful daily info. People have added other things to their builds, though, such as nice compliments or seasonal themes.

How can I get a magic mirror?

Build one

You can't really buy pre-made, Pi-powered magic mirrors at the moment. In fact, we haven't even seen kits that let you do it yet. However, there are plenty of build guides you can find, such as our big build in issue 54 (**magpi.cc/54**).

Software

Once you've built your mirror, you'll need to put some software on there so it actually works. If you head to **magicmirror.builders**, you can get software created by Michael Teeuw, who is one of the pioneers of Pi-powered magic mirrors. It's modular, so you can add your own info blocks to it to truly customise your mirror.

▲ Build your own magic mirror with a screen, frame, and coated glass

Commission one

If you don't have the tools or skills to make your own, then find someone who will make it for you. It will obviously be more expensive than building it yourself; however, it may end up being a bit better.

What's the difference between this and an infoboard?

To mirror or not

An infoboard will display the same sort of data as a magic mirror, but it's not a mirror. Because of this, people tend to build infoboards a little smaller than magic mirrors, which means you can put one in a more accessible and useful place such as a kitchen.

Infoboard kit

As you don't need to do anything, like create a frame to make the infoboard more 'natural' looking, you can get a large touchscreen kit for a Raspberry Pi and just mount it on your wall. That way, it also visually looks like it's not just a part of the usual furniture, but something for guests to investigate and even use.

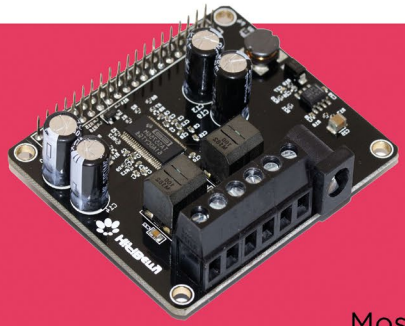
Browser habits

One important advantage an infoboard has over a magic mirror is that you can also use the browser properly in one. This is why we think it's good for the kitchen – you can easily use it to look up recipes – or check calories.

NEED A PROBLEM SOLVED?

Email magpi@raspberrypi.org or check raspberrypi.org/help for common issues. You can also find us on raspberrypi.org/forums

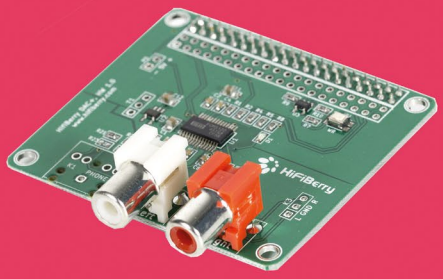
HiFiBerry
High Performance Audio For Raspberry Pi



HiFiBerry AMP2
integrated Digital-to-Analog Converter
60W power, up to 8Ohm speakers
supports sample rates from 44.1 - 192khz

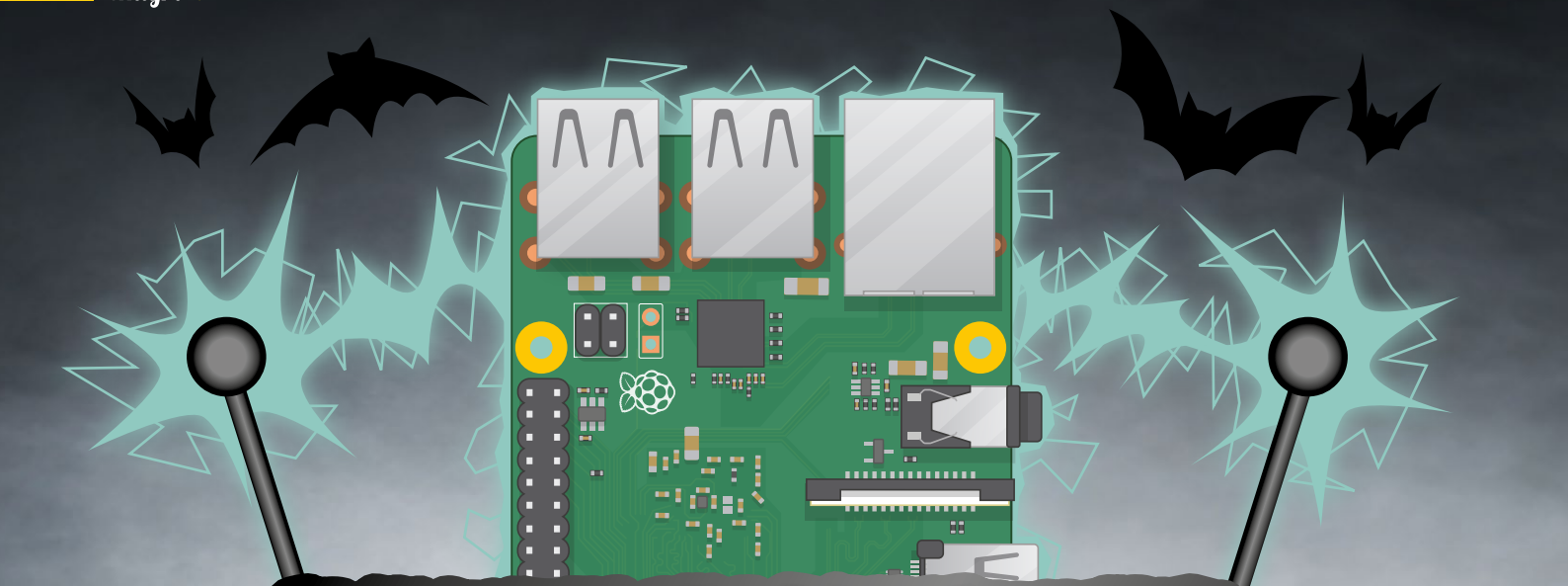
HAT modules
no soldering
needed

HiFiBerry DAC+
Most sold Digital-to-Analog converter
192khz/24bit offers high end sound quality
also availabe as DAC+ light, DAC+ pro and DAC+ DSP



What to built with HiFiBerry soundcards?
Small but versatile players with network based sources and streaming solutions, multiroom setups, upcycling of vintage speakers, DSP applications and many more.

visit hifiberry.com for complete product range, guides and inspirations



HAUNTED HALLOWEEN HACKS

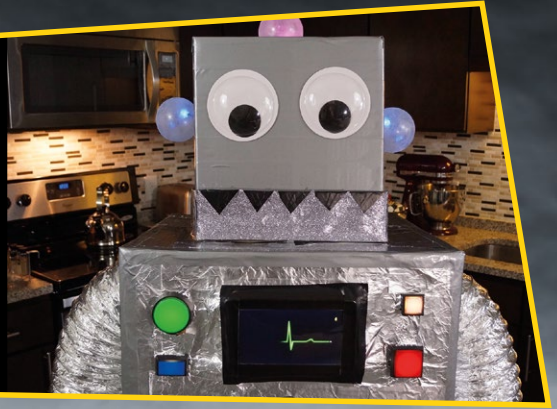
CREATE SOMETHING GRAVELY GOOD WITH A RASPBERRY PI

Welcome, foolish mortals, to this year's *MagPi* Halloween feature. We are your hosts, your ghost hosts. Kindly make sure we have your full attention as you read this. There's no turning back now.

As the air grows deathly cold outside, it can only mean one thing: All Hallows' Eve is almost upon us. A nightmarish night of abhorrent apparitions and torturous tricks. We're sure you'd love to be the repugnant ruler of shocking scares, and if you didn't think we'd put together a list of ghoulish guides for you this year you'd be dead wrong.

Oh, we didn't mean to scare you prematurely. The real chills come later. Now, as they say, 'look alive', and we'll continue through the next few pages of positively petrifying projects.

WICKED WEARABLES
PAGE 70



DEADLY DECORATIONS
PAGE 72

DREADFUL DOLLS
PAGE 74

MORBID MARATHONS

THE EASY WAY TO PUT ON A VISCERAL VIEWING PARTY OF YOUR FRIGHTFUL FAVOURITES

A good Halloween party should always have something horrifying playing in the background – and we don't mean *Batman v Superman*. You can easily set up Kodi on your Raspberry Pi using LibreELEC (**libreelec.tv**) and create a playlist of your favourite films.

Alternatively, why not create a terrifying alternative to the Pi Zero Simpsons Shuffler (**magpi.cc/FcbPwM**) that plays *Treehouse of Horror* episodes at the touch of a button? A perfect way to take a brief break from trick and/or treating.

▲ *Treehouse of Horror V* is definitely the best one

WICKED WEARABLES

A GOOD HALLOWEEN COSTUME REQUIRES SOME GRIM IMAGINATION

STEAMPUNK TENTACLE HAT

> magpi.cc/iqDuUm

It's good to stick to a theme, and going to a steampunk gathering wearing a creepy top hat with wriggling tentacles coming out of the top is a great example of this. Creator Derek Woodroffe makes this hat look simple, but it's actually a little more complex than it might seem at first.

"The base of the tentacles are powered with a pair of servos," David explains in his blog. "Each servo is connected to two pieces of nylon fishing twine. Each servo controls a degree of freedom, so one does X and the other Y."

With four tentacles, that's eight servos being individually controlled by a Raspberry Pi, in pairs, to move them independently and differently from each other. This is done via PWM, and uses sine waves to create a natural look. You can read more on how he did that on his blog.

The tentacles are pretty power hungry, though, and the required batteries couldn't fit into the hat. A special box was made to carry around with the hat, which gives it the necessary 'life support'.

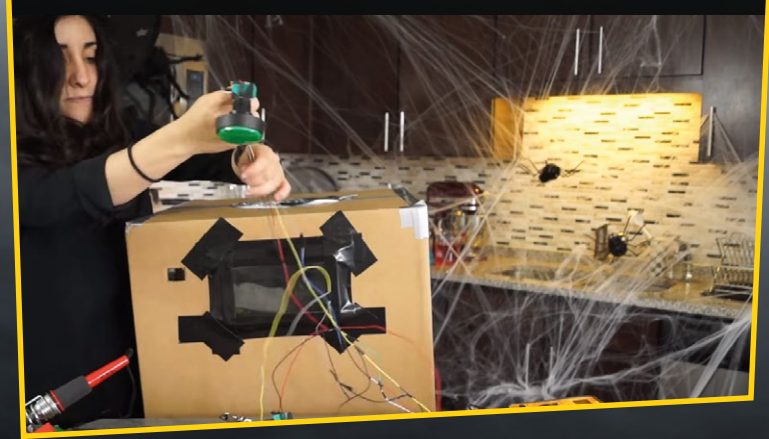
ROBOSUIT

> magpi.cc/HPmPZx

Is this an unstoppable, time-travelling assassin robot that all those films told us about?! No, it's just Estefannie in a Pi-powered cardboard box coated in aluminium. This excellent costume is pretty simple to make and includes voice changing, interactive buttons and screen, and blinking bulbs on the side of the head.

"To make my robo-suit come to life, I wrote a Python script that runs on a Raspberry Pi to check likes on my Instagram," says Estefannie. "Every time someone likes a picture on my Instagram, the script plays an electrocardiogram heartbeat video on the Raspberry Pi display."

The buttons control the LED colours, play random sounds through the voice changer, and can also start videos playing on the screen in the chest. It uses a mixture of Raspberry Pi (for the screen) and Adafruit controllers (for the voice stuff), and you can watch a full video of Estefannie creating it if you want to attempt your own: **magpi.cc/GbaLEh**.



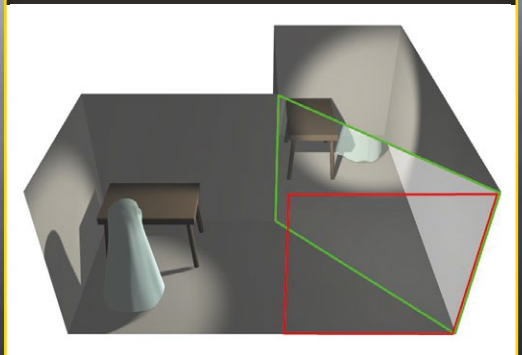
▼ Mounting the parts is less about screws and more about tape

TERRIFYING TECH TRICKS:

PEPPER'S GHOST

This classic stage illusion allows you to make a ghostly apparition appear in real life. It's still used today in theme park attractions (such as in Disneyland's Haunted Mansion) and is also the same tech for creating 'holograms' (think Hatsune Miku and Tupac) and uses angled glass to reflect an object placed elsewhere. The reflected image then takes on a ghostly, semi-transparent quality.

We love this video by Hackaday explaining how to perform the illusion: **magpi.cc/mhHHFm**. Maybe you can use it for a Halloween decoration or spooky photography?



▲ Lighting is key to getting the illusion to work

Image credit: Wapcaplet CC BY-SA 3.0

DEADLY DECORATIONS

SCARE YOUR FRIENDS WITH THESE **HOME HAUNTS**

▶ The ghastly visage activated by unsuspecting passers-by

POSSESSED PORTRAIT

> magpi.cc/2yMfQri

What better way to set a foreboding atmosphere in your house for Halloween than a creepy painting? Especially if the portrait is haunted itself – Dominick Marino managed to summon a poltergeist to inhabit his painting, which moves and screams at guests as they walk by.

> "IT MOVES AND SCREAMS AT GUESTS AS THEY WALK BY"

This particular painting is actually not a painting, but a monitor cleverly disguised as one thanks to an ornate-looking frame that's attached to it – much like some magic mirror projects! A Raspberry Pi hides behind the mirror, displaying a static image of the noble lord. A PIR sensor not-so-sneakily poking out of the frame detects any movement, which triggers VLC to play the full scary video.

The project has recently been updated so that you can also use it to prank your friends, and record their reactions via a Raspberry Pi Camera Module. Devilish indeed.

> magpi.cc/LaRYPA

It's creepy enough as a Halloween decoration without everything else

HALLOWEEN-OF-THINGS

Somehow society has managed to make clowns extremely scary, and we largely have Stephen King to blame for it. In this case, a very scary clown has been deposited on a front porch along with the rest of R Scott Coppersmith's Halloween paraphernalia. It's jarring, but it fits right in with the rest of the décor.

Except, this clown will sense trick-or-treaters coming to the door. Its eyes will light up and it will make strange and spooky noises at any unsuspecting kids in costume. To perfectly capture their face of fear, a photo is taken by a webcam and emailed to whatever address you wish. It might also prove useful as a makeshift security camera.

This build uses a clown mask from a Halloween store; a repurposed, motion-sensing LED light with the lights repositioned to the eyes; and a Raspberry Pi to play the noises and capture the moment of murderous terror from the victim.

TERRIFYING TECH TRICKS:

INFINITY MIRROR

An infinity mirror uses lights and mirrors to give the illusion that it stretches out to infinity. They're not too difficult to make, and the effect is amazing – you can even put objects in it that will then seemingly stretch into forever. Maybe a gruesome ghostly image trying to escape from the underworld?

There's a great tutorial on Instructables on how to make one with an IKEA frame (**magpi.cc/ZLYnNA**). You could upgrade it to use a Raspberry Pi and NeoPixel lights to give it a more interactive and spooky feel.

This circular mirror (**magpi.cc/xmALZh**) has a hole in the bottom – a great way to scare your friends

DREADFUL DOLLS

APPROACH A NEW LEVEL OF TERROR WITH THESE **FRIGHTENING FIGURINES**

REMOTE-CONTROLLED BILLY FROM SAW

> magpi.cc/NrLcAk

We're on something like *Saw 26* now, so clearly the franchise is extremely popular. Billy is a doll in these films that the bad guy uses to communicate with his victims, and it's pretty creepy. Instructables user David0429 decided to make his own Billy, complete with tricycle to ride around and scare the utter bejesus out of millennials.

The build uses an actual children's tricycle, cleaned up and repainted to look like the version in the film. He then 3D-printed custom gears so that the electric motor could actually drive the tricycle. Wires are then run through the frame, up into the seat, where a Pi awaits to control them via the remote control synced up to it. A steering servo is wired up to move the front wheel so it can steer – also giving the illusion that the handmade Billy puppet is moving the handlebars as it turns.

> "IT SPOOKS US A BIT BUT WE HAVE TO ADMIT, IT'S AN EXCELLENT BUILD"

It spooks us a bit but we have to admit, it's an excellent build.

▲ We're not sure if this is more spooky or not?

▼ Electronics cleverly hidden in the saddle

TERRIFYING TECH TRICKS:

HAUNTED MIRROR

The idea of this is a bit similar to a haunted portrait – this one uses the popular magic mirror concept and adds a ghostly video which can activate when somebody walks past. You could have a spooky character appear in order to scare them, or even something like a train rushing towards the mirror.

We like this Instructables build here: **magpi.cc/HQZLHg**. It uses the same magic mirror style stuff a lot of people use a Raspberry Pi for, so it's easy to adapt your own magic mirror if you have one. Experiment and see if you can make a murderous mirror worthy of creating screams loud enough to wake the dead.

▲ You can even create a spooky theme for your normal magic mirror, like Bradley Melton did in issue 40 of *The MagPi*

SCARY DOLL

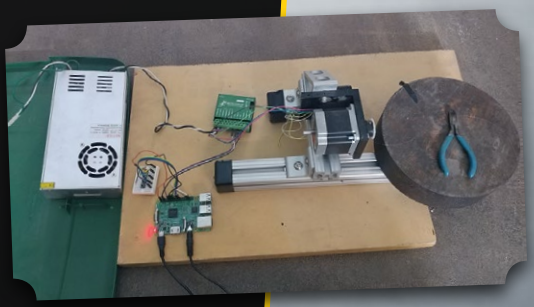
> magpi.cc/FXiBMC

Billy isn't the only scary doll in films; there's also Chucky and that evil Krusty doll. Cabe Atwell tries to explain why he finds them scary:

"My fear must stem from a movie I saw as a child, called *Dolls*. It frightened me so bad, I literally could not sleep, not even in the day! No other film did that to me. The Chucky series, *Goosebumps* episodes with the ventriloquist puppet, none of these scared me as a kid. It was something about that movie, *Dolls*... fuel for nightmares."

This aptly named Scary Doll is designed to float off the ground using a hidden motor and some fishing wire. Excellent for a practical video effect, maybe a little too obvious in broad daylight. Best to keep it for a dark night full of trick-or-treaters.

The Raspberry Pi controls the pulley, and it can also be used to spin the doll around so that it seems a little more... (re)animated. It would definitely give us a scare if it jumped out of a box at us.



▼ The motors and servos used to levitate and move the doll

▼ Hiding it in a box is a good way to scare some unsuspecting folks

Picade

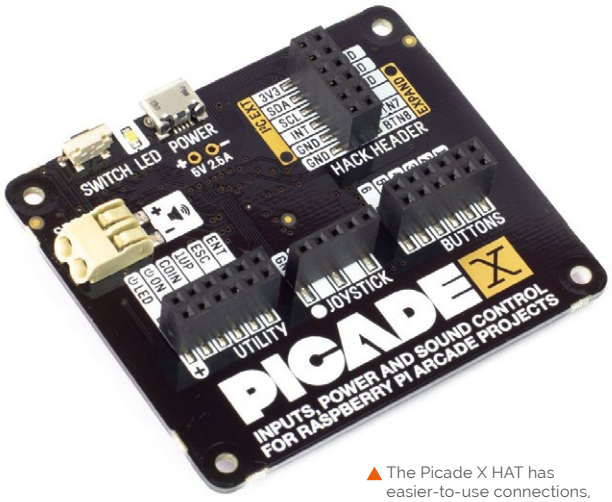
▶ Pimoroni ▶ **magpi.cc/iLOfHv** ▶ £150 / $161

The new Picade model is sleeker with a host of improved features. **Phil King** relives his misspent youth down the arcades

New versions of products are usually billed as 'bigger and better', but Pimoroni's new Picade is smaller than its original mini-bartop arcade cabinet. The display is still 8 inches, however, lacking the thicker black surround of its predecessor. This time it's an IPS (in-plane switching) panel with wide viewing angles, higher resolution (1024×768 compared to the earlier 800×600), and a new Pimoroni-designed driver board with HDMI input and keypad controls for an on-screen menu.

Another key improvement is the new Picade X HAT, which works with any 40-pin Raspberry Pi. Also available separately (£15) for those who want to build their own custom arcade cabinet, the HAT has easy-to-use DuPont connectors for the numerous joystick and button wires. An additional 'Hacker' header breaks out the few remaining unused GPIO pins and I²C, which could be used to add extra buttons. The HAT also features power management and a built-in I²S DAC with 3W amplifier for mono audio – this time there's only one speaker included, although it's plenty loud enough.

◀ The new Picade is easier to build and looks fabulous sitting on your coffee table



▲ The Picade X HAT has easier-to-use connections, including a 'Hacker' header

Before you play on your new Picade, you'll need to assemble it. Taking 2–3 hours, this is an easier process than before, although there are still fiddly bits – mainly involving holding the tiny M3 nuts in hard-to-access places while screwing bolts

> Before you can play on your new Picade, you'll need to assemble it

(tip: use Blu Tack). Full instructions are supplied on the reverse of an A1 poster, but we found the appended online ones, with videos, easier to follow. Assembly is aided by some excellent packaging, with separate colour-coded boxes for the cabinet, screen, fixings, and accessories.

Arcade assembly

Firstly, a few of the black powder-coated MDF panels need to be screwed together with plastic brackets. Placed upside-down onto a clear acrylic panel, the screen display is connected to its rear-mounted driver board by a short flat flex ribbon cable and care needs to be taken not to pull out the connector tabs too far when inserting it.

Next, add the 30 mm push-fit arcade buttons and a microswitched joystick with ball top. Since these are standard parts, you could potentially customise your Picade by using different (possibly illuminated) buttons and joystick topper.

The wiring is easier than on the original Picade due to the DuPont connectors on the HAT, so you simply push in the pins of the wires, although the other ends still have spade connectors (and there are push-fit connectors for the speaker wires). As long as you get each wire loom the right way round at the HAT end, you should be able to make the correct connections for the joystick and buttons. In addition to the six control buttons, there are four utility buttons placed around the cabinet and a light-up yellow power button – simply press this to turn the Picade on and off, automatically shutting down the Raspberry Pi safely – a really nice touch.

Playtime

Before turning on, you'll need to download RetroPie and write it to a microSD card – and uncomment a line in the **boot/config.txt** file to force HDMI output, to make the display work. The card can then be inserted into the Pi mounted inside the cabinet via a handy access hole. Alternatively, the back panel can easily be removed for easy access to all the components.

A standard 5 V micro USB power supply (not included) powers the Picade X HAT, which in turn powers the Pi, and the display via a USB cable. Hit the power button and away you go. Well, not quite. You'll first need to connect a keyboard to the Pi and install the Picade X HAT driver with a one-line Terminal command.

Then it's just a matter of setting up the joystick directions and buttons in the EmulationStation menu and – after adding ROM files to RetroPie – playing your favourite games. *Hadouken!*

◀ So, what will it be first? Popeye? Donkey Kong? Street Fighter?

SPECS

SCREEN:
8-inch IPS panel, 1024×768 pixels

BOARD:
Picade X HAT

CONTROLS:
Joystick, 6 × arcade buttons

SPEAKER:
3-inch, 5 W, 4 Ω

DIMENSIONS:
350 × 230 × 210 mm

Verdict

A fun, if at times fiddly build, this all-new Picade features high-quality components and feels sturdy. Major improvements over the original version include a vivid, higher-res IPS display and easier-to-connect Picade X HAT.

9/10

▶ Konichiwakitty ▶ **magpi.cc/dwNcHG** ▶ £22 / $28

SPECS

LIGHTS:
8 × multicolour LEDs

POWER:
3 V, battery operated

FLUFFY YARN:
6 m, varying colours

Introduction to Soldering: Kitty Ears

This cute, wearable kit is designed to get girls interested in electronics through the power of LEDs and cats. **Rob Zwetsloot** is feline up to the challenge

Everyone should be able to enjoy coding and making, wherever on the gender (or indeed any) spectrum they may be. The Raspberry Pi Foundation, and we on *The MagPi*, are working on bringing down cultural barriers so young women can start their digital making journey. Products like this kit – featuring cute yarn and awesome LEDs – are one way to help effect this change.

Many products aimed at young girls can feel patronising and dumbed-down, but despite appearances, this is a legitimate introduction to soldering: you will absolutely need a soldering iron to build this, and there are no short cuts. Once it's all connected, you get the gratification of the circuit working – an instant, and positive result!

The rest of the build is simple, and involves wrapping yarn around a plastic headband. The process teaches you about common knitting knots, and allows customisation by adding extra trinkets and accessories to truly make the ears your own.

Excellent instructions

The box comes with extremely thorough instructions, and there are detailed video walkthroughs on Konichiwakitty's website (**magpi.cc/rtGZPj**) which show you every step of the process. The components that make up the headband can be used on different builds as well – by adding more LEDs you can make it brighter, or attach it to different headbands with different ears.

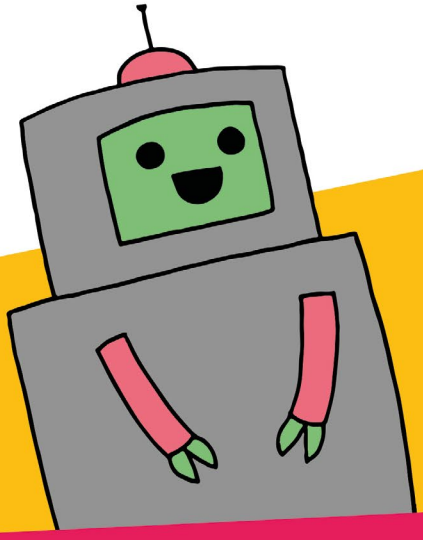
It is a fantastic kit, overall. While not all girls like pink cute things, some do (and shouldn't be shamed for it), and it's a fun afternoon for anyone who decides they'd like to build one. And Halloween is right around the corner…

▲ Not all girls are into pink and fluffy kittens… but an ever-growing number love a bit of soldering action

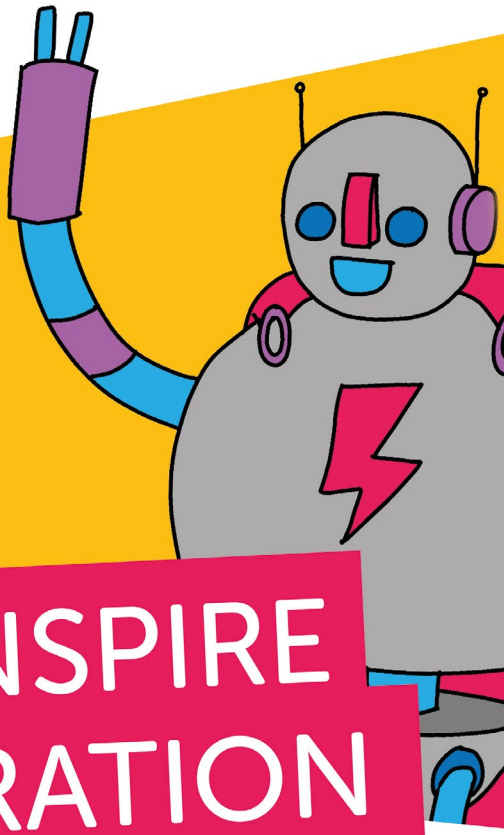
Verdict

A great kit that can truly help some people feel like they can become a maker. With plenty of detailed instructions and videos available, it should be a fun afternoon build.

8/10



{code club}

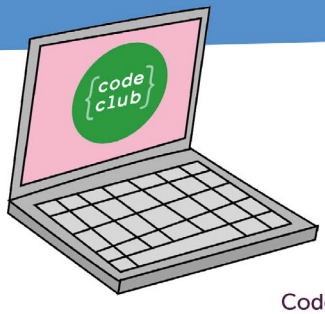


CAN YOU HELP INSPIRE THE NEXT GENERATION OF CODERS?

Code Club is a network of volunteers and educators who run free coding clubs for young people aged 9-13.

We're always looking for people with coding skills to volunteer to run a club at their local school, library, or community centre.

You can team up with friends or colleagues, you will be supported by someone from the venue, and we provide all the materials you'll need to help children get excited about digital making.



To find out more, join us at

www.codeclubworld.org

Code Club is part of the Raspberry Pi Foundation. Registered Charity Number 1129409

10 Best:

Raspberry Pi cases

Keep your Raspberry Pi safe with one of these cases

While it's actually pretty tough, it's always best practice to put your Raspberry Pi into a case. This helps protect it from many hazards and accidents, and can help it blend into its surroundings as well. Plus, a lot of cases available just look rather cool. Here are ten of our favourites.

Official case

TYPE: Officially functional

This lovely case sets a basic standard for all Raspberry Pi cases. It snaps together around the Raspberry Pi, allowing you to fully encase it if desired. However, it's also incredibly easy to take the top layer off to access a HAT, or the GPIO side bit to access those pins. It's also sleek and looks great.

- £6 / $8
- **magpi.cc/TCxsXH**

FLIRC

TYPE: Stylish heatsink

FLIRC is well known for the customisable IR receivers it makes for Raspberry Pi media centres, but it also makes this gorgeous case. Not only is it designed to look good and fit seamlessly into a high-tech TV setup, it also functions exceptionally well as a heatsink for the Raspberry Pi.

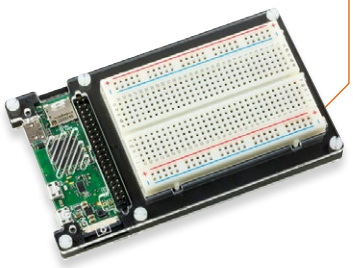
- £15 / $19
- **magpi.cc/cJkoah**

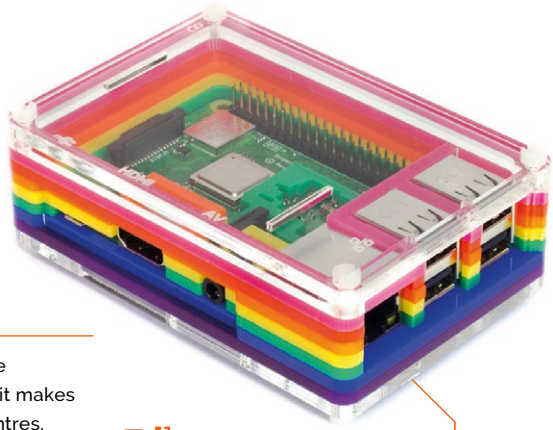
Pi Zero Breadboard Case

TYPE: Easier prototyping

While not all Raspberry Pi boards are used for prototyping circuits or learning about electronics, when you are actually doing that it can be a bit tricky to then move your project elsewhere. Not only does this case help you transport your breadboard project a bit more easily, it looks smart and actually makes wiring up electronics a touch easier.

- £10 / $13
- **magpi.cc/LkvKAD**



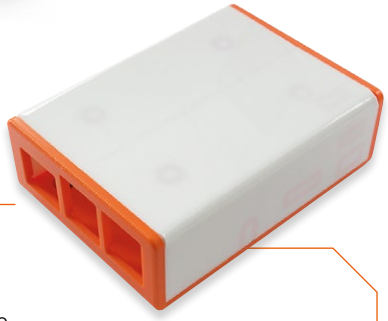


Pibow

TYPE: Laser-cut slices

The original Raspberry Pi case that you didn't have to make out of LEGO, Pibows have changed a lot over the years, but they're still made up of precisely cut acrylic slices that you slide over the Raspberry Pi one at a time to build up into a full case. There are many colours to choose from as well.

- £13 / $17
- **magpi.cc/QoaJAa**



Flick! HAT

TYPE: Gesture control

The Flick! HAT is a gesture control input for the Raspberry Pi. It's quite simple but can be extremely useful, and this case is designed specifically to work well with the HAT. There's even a Pi Zero version. While the Flick! HAT will work through the top of the acrylic, you can open up the top of the case, and even punch holes in it to have better access to GPIO pins and such.

- £10 / $13
- **magpi.cc/pfJPGp**

Super Kintaro

TYPE: Retro cooled

At first glance this retro-inspired case is pretty cool, even if it's based on the SNES from the wrong side of the Atlantic. It's perfect for your retro gaming setups, especially if you missed out on the SNES Classic. It includes a massive heatsink (sadly incompatible with Pi 3B+), and you could even fit a tiny case fan inside for extra cooling powers.

- £20 / $26
- **magpi.cc/bidPmE**

Nucleus Wood Zero

TYPE: Natural look

This beautiful Pi Zero case has both form and function. It's made from mahogany, and offers excellent ventilation for the board. While you can't really access the GPIO pins with it on, it's still a lovely case that properly protects the Pi Zero inside.

- £4 / $5
- **magpi.cc/QiHrgS**

ZeroView

TYPE: Window view

The ZeroView is a very simple case idea. Insert your Pi Zero, connect a Pi Camera Module, and then use the included suction cups to attach it to a window. It's great for time-lapses of your garden, a motion-sensing security camera for your front porch, or even a dashcam for your car. We love it.

- £7 / $9
- **magpi.cc/2e89hWt**

LCD Case

TYPE: Compact screen

The 3.2-inch screen form-factor is very popular for Raspberry Pi screens, and SB Components makes an excellent case which fully exposes the screen (and the usual extra buttons), while keeping the rest of the Pi enclosed. It's useful for several project types, and overall just looks nice and compact.

- £4 / $5
- **magpi.cc/VTypXZ**

PiShell

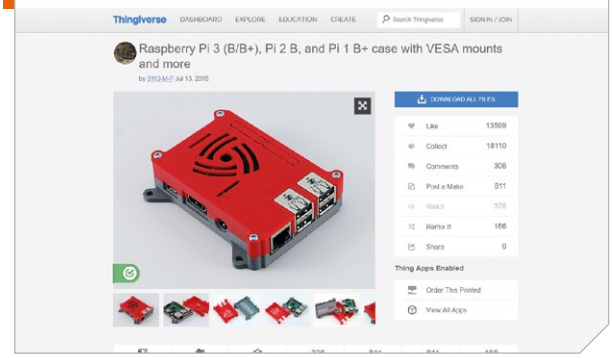
TYPE: New configuration

Recently crowdfunded, the PiShell should be available shortly after you read this. It's a good and sturdy case that, similarly to the official case, can be partially taken apart to access the top of the Pi. Unlike the full-size official case, it has a wall-mounting point and a hole for the camera to peek out of as well.

- £6 / $8
- **magpi.cc/ZZcTZC**

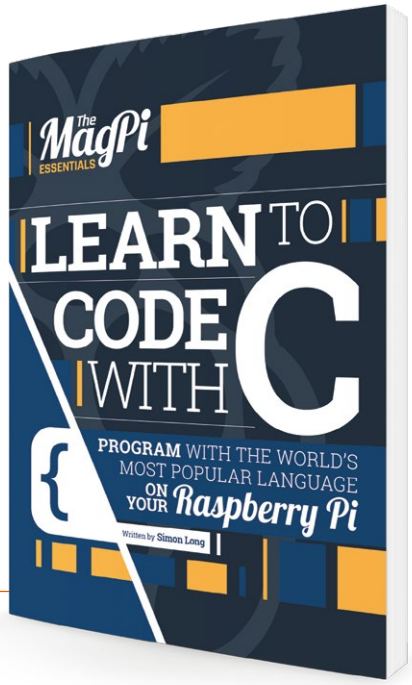
CUSTOM CASE

These cases not what you want? Then you can always 3D-print your own Raspberry Pi case. If you're not up to designing one yourself, make sure to check Thingiverse: **thingiverse.com**



Learn C with Raspberry Pi

Lucy Hattersley looks at the best resources for learning the venerable C programming language



Learn to Code with C

AUTHOR
Simon Long

Price:
£3.99 (Free download)

magpi.cc/learn-c-book

If you want to learn to program C using a Raspberry Pi, then the best place to start is with our official guide.

Written by Raspberry Pi's very own Simon Long, the UX Engineer responsible for creating the Desktop on Raspbian, this book is the perfect guide to C on the Raspberry Pi.

Simon has been programming with C since the early nineties and has created everything from mobile phones to medical equipment. The result is a compact and practical programmer's guide, written from direct experience.

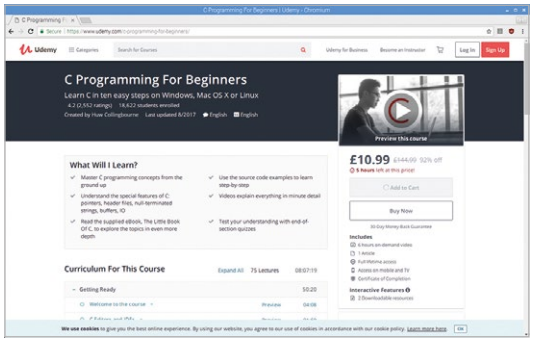
Starting with Hello World, the book covers everything from basic variables, strings, and arithmetic, through pointers (a subject many find difficult), and right up to file input and output.

What's great about the *Essentials Guide* is that it's a pocket-sized guide to the C language, packing in all the basics in 92 pages. Unlike many books on C, which run to thousands of pages, you get all the essential information and very little waffle.

Many members of *The MagPi* team find this book an invaluable reference guide, and keep a copy on hand whenever programming in C.

C courses

Learn C by following an online course


C Programming For Beginners
Learn C in ten easy steps on Windows, Mac OS X or Linux
What Will I Learn?
Curriculum For This Course
£10.99

▲ Look out for special offers on Udemy's beginner-friendly course

C PROGRAMMING FOR BEGINNERS

Udemy's course, by Huw Collingbourne, is extremely beginner-friendly and provides easy-to-follow video tutorials. There's a huge range of content. The RRP is £29.99, but you will periodically find it on sale (at press time it was £10.99).

▶ magpi.cc/rLOorl

PROGRAMMING PARADIGMS (STANFORD)

This classic series of lectures by Stanford Professor Jerry Cain can be found on YouTube. The course teaches students how to write several programming languages, including C, and how to understand the programming paradigms behind each language.

▶ magpi.cc/KtTfjS

INTRODUCTION TO PROGRAMMING IN C

This four-week course by Duke University is designed to teach problem-solving with the C programming language. It's not cheap (a subscription to Coursera is £37 per month), but Coursera certificates are recognised by many companies, so it's a good option for those looking for a professional qualification.

▶ magpi.cc/SHQtVy

CS50's Introduction to Computer Science

AUTHOR
David J Malan

Price:
Free

magpi.cc/KMEpHx

Massive open online courses (MOOCs) are online training schemes available for anyone to enrol in. We're big fans of MITx, but Harvard has hands-down the best introduction to Computer Science with the C programming language.

CS50 starts with Scratch and Python, but quickly switches to using C in a custom IDE as the basis for explaining computer concepts. It also features other technologies such as SQL and HTML.

Harvard says: "CS50x teaches students how to think algorithmically and solve problems efficiently. Topics include abstraction, algorithms, data structures, encapsulation, resource management, security, software engineering, and web development."

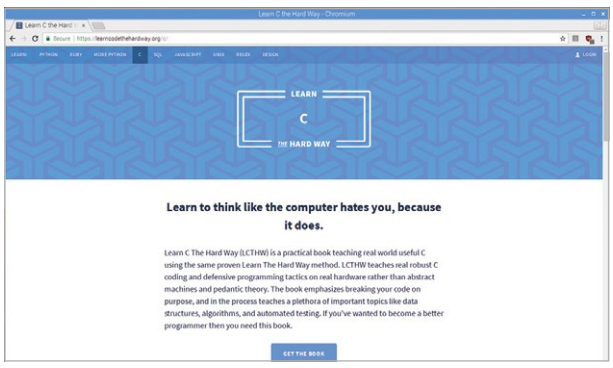
CS50 is a challenging course, and towards the end you'll have a comprehensive understanding of C. It's a fantastic way to move from Scratch and Python to advanced coding in C.

Learn C The Hard Way

AUTHOR
Zed A Shaw

Price:
$29.99

magpi.cc/Tizkpp



Few things divide the coding community as much as Zed's programming courses. His ruthless, no-nonsense approach to learning eschews niceties such as developer environments and online interactive websites, instead throwing you into a text editor and the terminal.

This frank acknowledgement that programming "is hard" and that visual fluff distracts newcomers from actually learning is just the cold-water shock many programmers need. And nowhere is his course in its element more than when learning C (where you need to learn how to build, make, and run source code).

You can follow the courses online for free, but paying the charge gets you screencast videos, a PDF book, and free updates for life. Your editor loved this course so much, she paid up just as a 'thank-you'.

Resources

Have these resources on hand

CPLUSPLUS.COM

C++ is a more advanced programming language built on top of C, and **cplusplus.com** is the definitive guide. Within it, though, you'll also find a great reference resource for C itself. Here you'll find detailed information on functions, containers, and input/output commands. Just don't get sidetracked by C++ until you're ready.

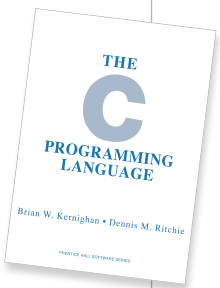
▶ magpi.cc/EZfPFn

STACK OVERFLOW

You should set up a Stack Overflow account as soon as possible, no matter what language you're learning. But this venerable site is especially helpful for getting answers to C questions. Use '[c]' in your search term to get C-tagged results.

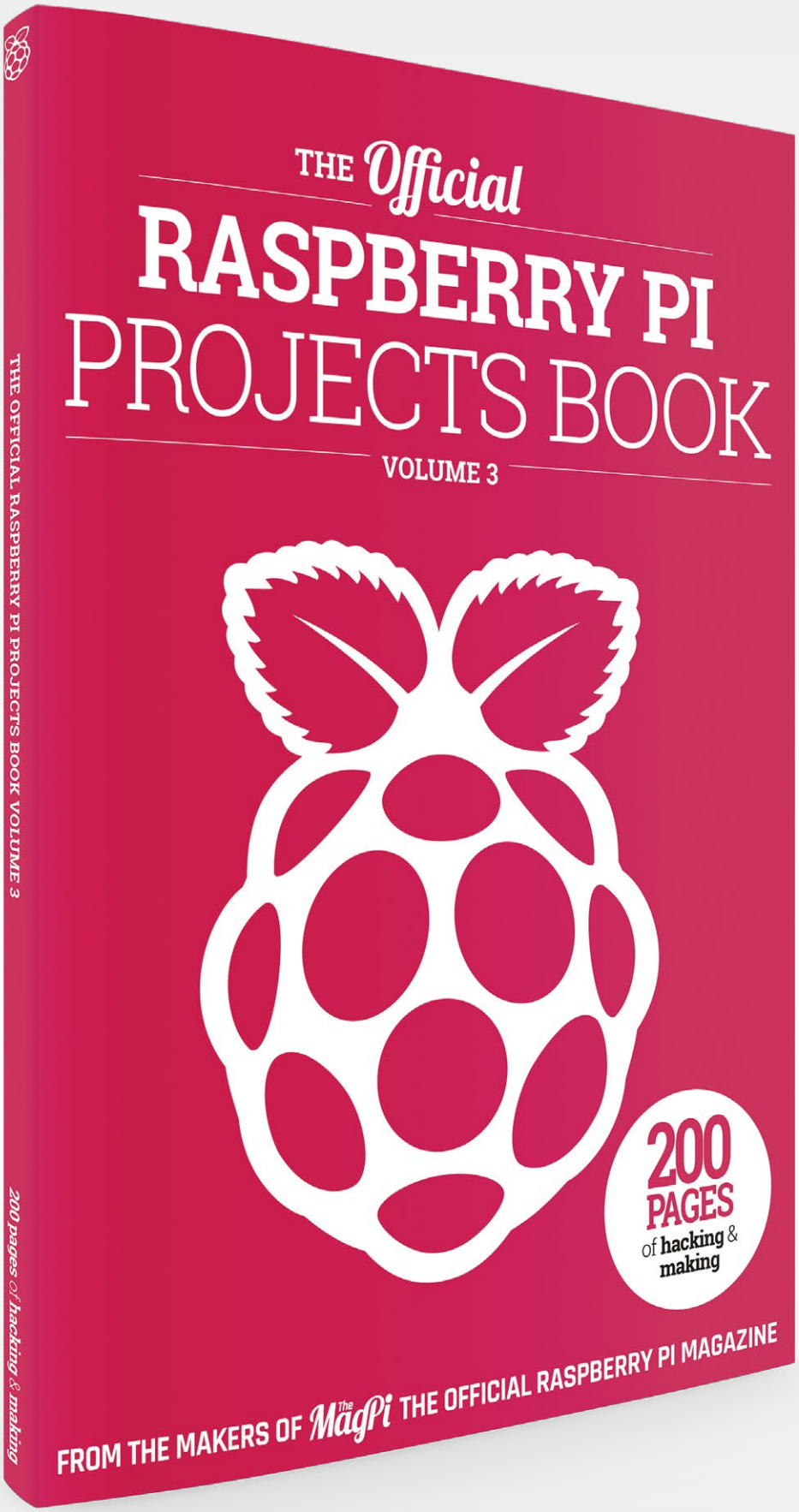
▶ stackoverflow.com

THE C PROGRAMMING LANGUAGE



This book by Brian Kernighan and Dennis Ritchie, the latter of whom originally designed and implemented the language, is the definitive guide. The book itself is heavy-going, and you're better off following a course when starting out. But it's very handy to have around.

▶ magpi.cc/CUQyBO


THE Official
RASPBERRY PI
PROJECTS BOOK
VOLUME 3
200
PAGES
of hacking &
making
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE
THE OFFICIAL RASPBERRY PI PROJECTS BOOK VOLUME 3
200 pages of hacking & making

£12.99
200 pages of
Raspberry Pi

THE Official
RASPBERRY PI PROJECTS BOOK
VOLUME 3

Amazing hacking and making projects
from the makers of The MagPi magazine

Inside:

- How to get started coding on Raspberry Pi
- The most inspirational community projects
- Essential tutorials, guides, and ideas
- Expert reviews and buying advice

Available now magpi.cc/store

plus all good newsagents and:

WHSmith BARNES&NOBLE

Available on the App Store

GET IT ON Google Play

Image Credit: Smokey Nelson

Becky Stern

DIY expert, maker, biker, and vlogger. Becky can apparently do just about anything

- Category Maker
- Day job Vlogger
- Website beckystern.com | magpi.cc/DFIfAT
- Twitter @bekathwia

Growing up with parents that love to do a bit of DIY can often rub off on kids – even if it's just knowing how to put a shelf up. For Becky Stern, this heavily imprinted on her, especially as she was recording the process.

"I can remember holding our gigantic VHS video camera to film the assembly of a big wall going up on the home addition they were building," Becky tells us. "I was probably five or six, and too small to help lift up the long bit of framing."

She started her making career at age eight, by sewing together a 'knock-off' of a Beanie Babies toy. "Computers also fascinated me, and I spent a lot of time on BBSes, browsing the early net, and chatting with strangers on ICQ," Becky recalls. "I got into sewing my own purses and altering my clothes as a teenager. I was in AV club in high school, where I made short movies with my friends and learned how to use video editing software."

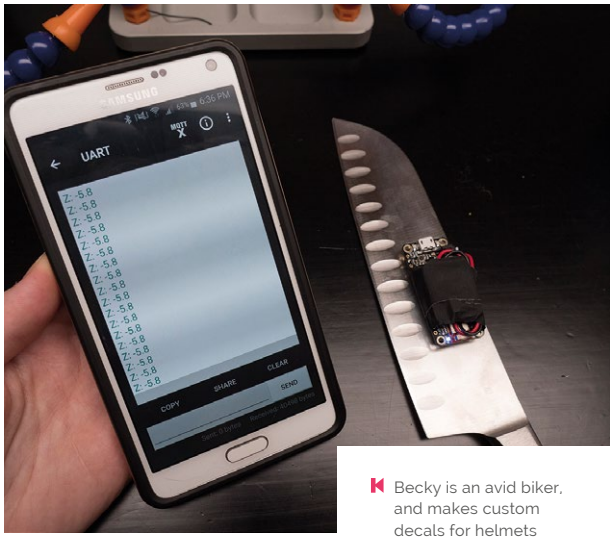
> "Follow your curiosity! Don't get caught up in having the exact right tools or materials"

▼ This simple build allows Becky to take photos of the things she's working on while her hands are otherwise occupied

After studying design and technology, she landed part-time gigs at Make: and Craft: magazines, before moving onto Make: full-time and finally Instructables. While doing that, she's been making amazing things, and releasing excellent videos about them.

Any dream builds you'd love to make?

I'd like to embed RFID tags in handmade fine jewellery. I'd like to make some floating bedside nightstands. I want to hook up a foot pedal to the intercom so the dog can buzz the door […]. I've had bad knees since I was a kid, so I tend not to imagine large or strenuous



Becky is an avid biker, and makes custom decals for helmets

Want to check to see how sharp your knife is? Becky has a project for that

projects outside my somewhat-limited physical abilities.

What hobbies and interests do you have outside of making?

I like to ride motorcycles, which definitely influences some of my builds, like my custom seat or helmet decals. I also love my pets, food and cooking, and playing video games. I definitely draw from my personal experience when brainstorming new project ideas, so naturally those other interests cross-pollinate my projects.

What would you say to someone who is thinking about starting to make stuff but hasn't taken the leap yet?

Follow your curiosity! Don't get caught up in having the exact right tools or materials, and let your available supplies help shape what you make at first. Take every chance to reduce the stakes of experimentation and failure. Brainstorm project ideas that will solve your small problems and annoyances […]. Design incremental successes for yourself, like finishing a project instead of adding that extra feature (at first). If you think of your learning curve as a video game, make sure it's fun to play and the levels are of appropriately increasing levels of difficulty.

Project highlights

TUMBLR GIF CAMERA

Usually, Becky says she's most satisfied with simple yet useful projects "Of course, complex electronics projects give me a strong sense of accomplishment (when they go well)," Becky says. "Like the vintage camera I upgraded with a Pi to create and upload GIFs to Tumblr."

▶ **magpi.cc/GLrUEb**

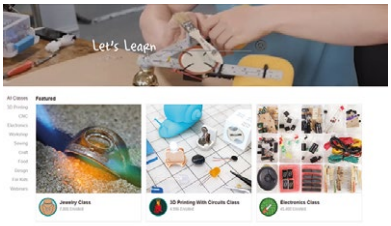
EMBROIDERED PHOTOS

"Projects with a process craft make me happiest while doing them," Becky reveals. While a fairly simple project, this is a great way to improve your embroidery skills and make something truly personal in the process.

▶ **magpi.cc/pYGafL**

LEARN TO MAKE

"If you're looking for the basic skills in all the common making disciplines, check out the free online classes at Instructables," Becky says. "I wrote the Arduino, Internet of Things, knitting, solar, and jewellery classes, and learned leather-working, robotics, and how to use a table saw by following my colleagues' classes."



▶ **instructables.com/classes**

This Month in **Raspberry Pi**

Picademy **Seattle**

Free teacher training continues in the US

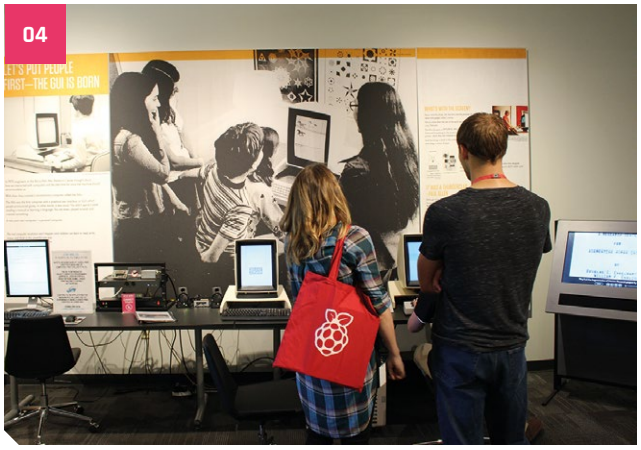
A Picademy event is very special, bringing teachers and educators together to get a better understanding of how to use the Raspberry Pi for computing education. The ones organised by the North American Raspberry Pi Foundation are no different, albeit covering a much larger area than the UK. There have been a few recent Picademy events, including the Seattle one that was held in the amazing Living Computers Museum.

"We trained 79 educators hailing from 17 US states and four different countries," says Andrew Collins, Raspberry Pi Educator Training Manager. "All came from variety of teaching backgrounds: makerspace leaders, teacher trainers, university professors, museum staff, classroom educators."

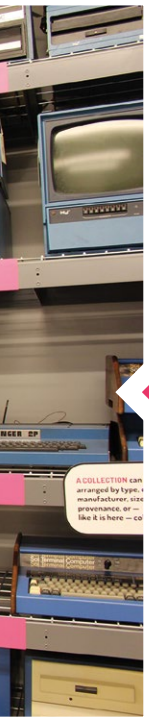
Here are some of the amazing photos from the weekend.

01. Racks of old computers
02. The museum is very thorough
03. There's lots of swag on offer
04. History adds context
05. Displays have changed a lot
06. Celebrating graduation

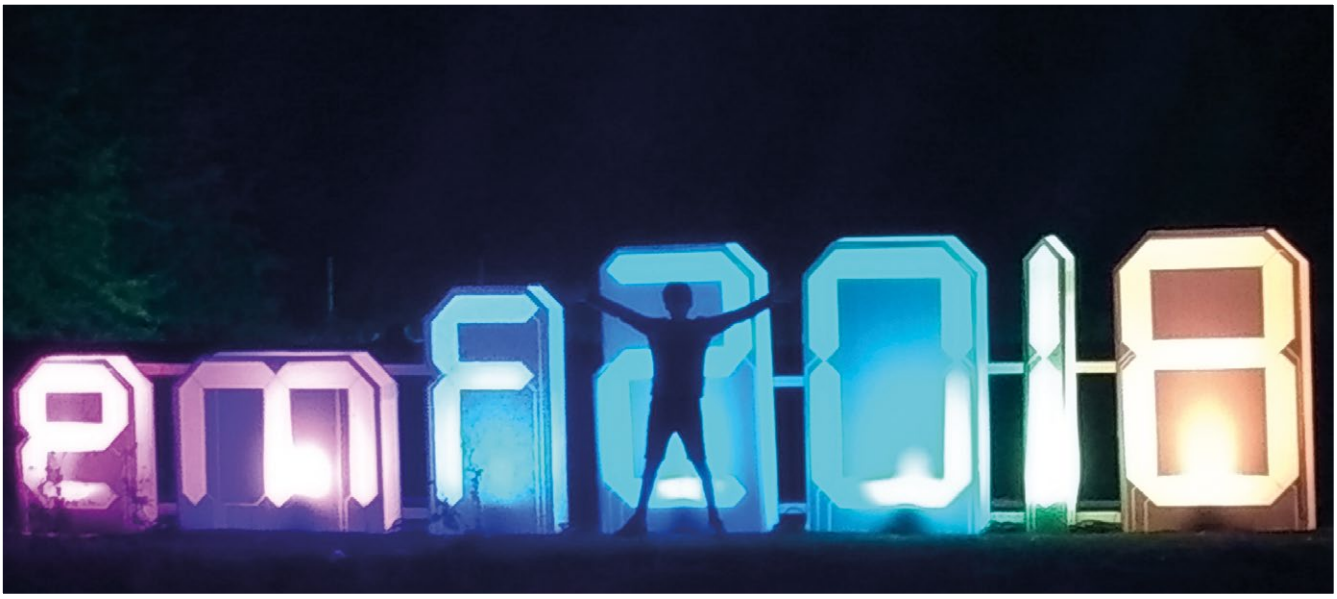
03
PARTS PACK
PICADEMY


04

05
OUR GOAL IS TO BREATHE LIFE BACK INTO OUR MACHINES



06



Electromagnetic **Field**

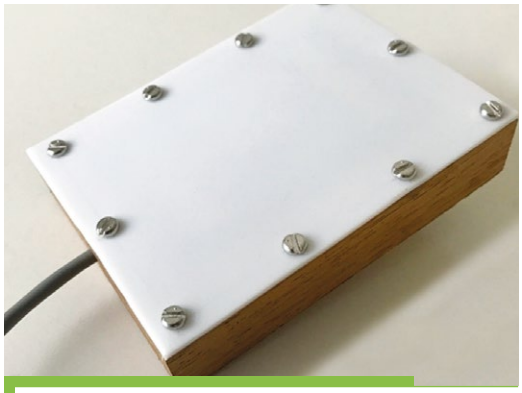
Usually, camping and electronics don't quite go together. The folks at EMF proved it could be done

We love a good maker faire, Raspberry Jam, or any fun event where people are showing off and having fun with tech and making. We never expected to see camping in that mix, but it's exactly what happened in Eastnor Castle Deer Park, Herefordshire, in early September. Even away from sheltered buildings with easy electricity access, UK makers managed to show incredible projects from robots to fashion wearables. Take a look...

01. A giant, workable ZX Spectrum
02. Beautiful views from above
03. Live view of the campers

Crowdfund **this!**

Raspberry Pi projects you can crowdfund this month



Solar Irradiance Meter

This is something we've never really thought about: while solar power is great, how do we know how much power the sun can provide at any given time? This Solar Irradiance Meter lets you measure it directly – it's not a solar power supply for your Raspberry Pi, though. We see this as an excellent bit of renewable energy IoT.

▶ **kck.st/2N0q4fM**

Useless Nixie Device

While it may be called a 'Useless' Nixie Device, don't let the name fool you. It is a fully functional Nixie tube, it's just that you could perceive a Nixie tube as useless. We don't think it is, though: we reckon they're cool and work in many project types. Anyway, this one lets you connect dozens of Nixie tubes to a Raspberry Pi and take control of them for numbers or simple graphics or whatever you fancy.

▶ **magpi.cc/oTxmrh**

CROWDFUNDING A PI PROJECT?

If you've launched an irresistible Pi-related project, let us know!

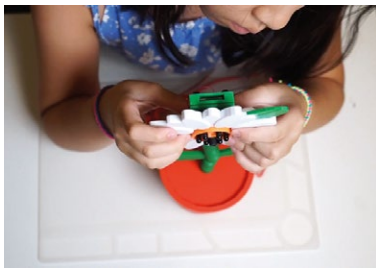
magpi@raspberrypi.org

Best of the rest!

Here are some other great things we saw this month

IR NIGHT VISION CAMERA

Purple Oranji sent us this great how-to video about how she made an infrared camera so she can stream video of her hamsters at night when they're active. You can follow along and make your own, especially if you want to keep an eye on a hamster.



▶ **youtu.be/tVmH1V4TBdY**

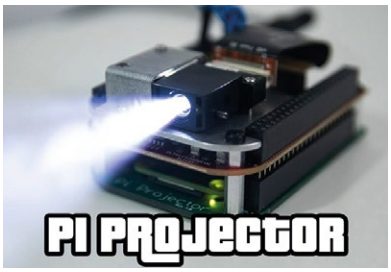
DIRTY PI: ANALOGUE VIDEO MIXER

This 'sound reactive' video mixer (and glitcher?) was created for user sealcouch's band. "The centre knob fades between the video playing on each Pi, while the side knob is a volume knob. The software is LibreELEC preloaded with random clips I pulled from YouTube."

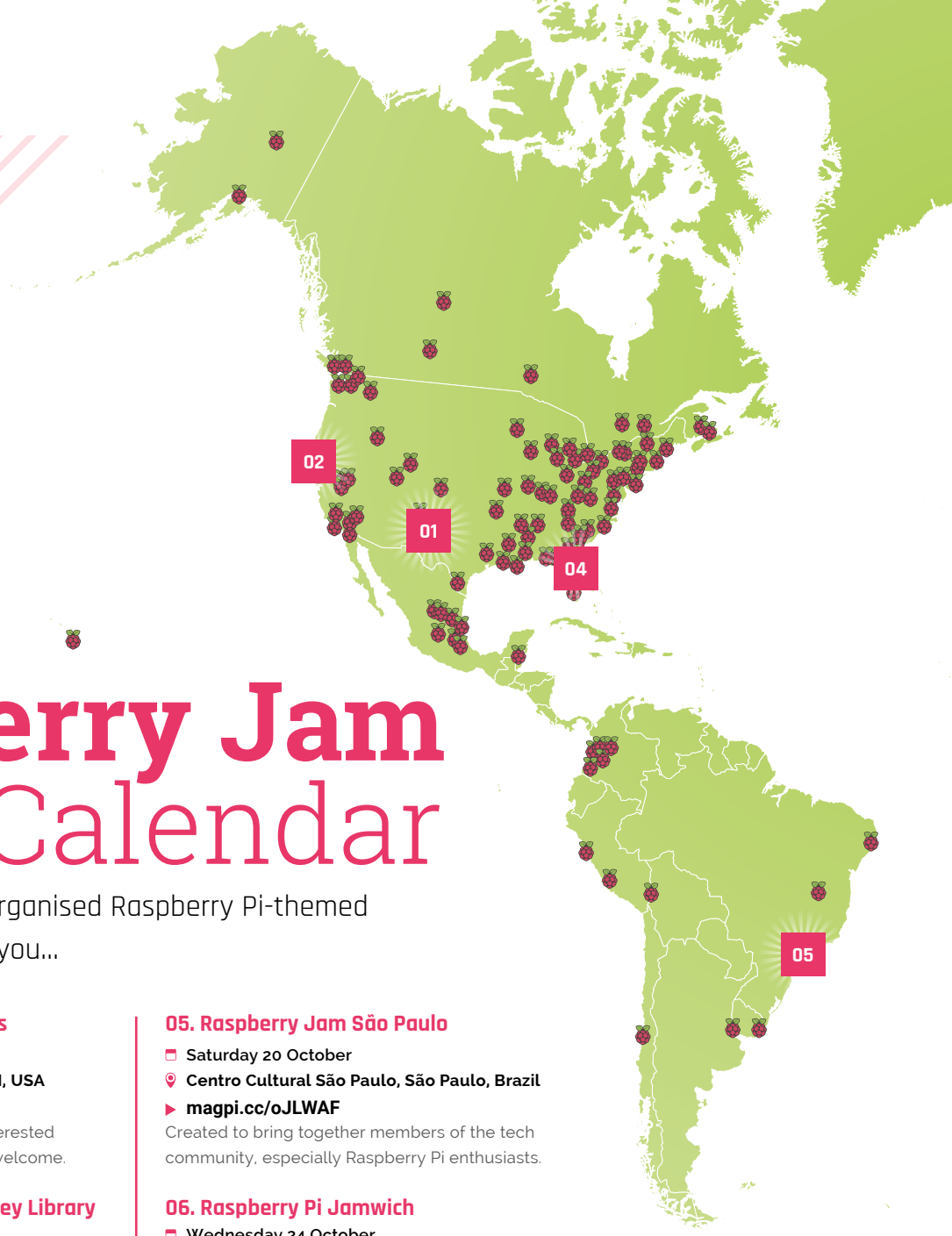
▶ **magpi.cc/DmKvvJ**

PI ZERO W PROJECTOR

Projectors are cool, and so are portable projectors. This project uses a Pi Zero to create one of the smallest and most portable projectors we've ever seen. It features a custom PCB created by MickMake, and while it doesn't do sound as is, you can easily add a USB sound card.



▶ **youtu.be/RxQ4GFfPJFo**



Raspberry Jam Event Calendar

Find out what community-organised Raspberry Pi-themed events are happening near you...

01. Raspberry Pi Club Los Alamos

- Thursday 4 October
- Los Alamos Makers, Los Alamos, NM, USA
- magpi.cc/EyPJsd

A brand new weekly club for people interested in the Raspberry Pi, where everyone is welcome.

02. Raspberry Jam @ Castro Valley Library

- Saturday 6 October
- Castro Valley Library, Castro Valley, CA, USA
- magpi.cc/KDdPLU

If you're interested in computer coding, here you can get involved in tinkering, coding, and electronics.

03. Next Tech Lab AP Jam

- Saturday 13 October
- SRM University AP, Mangalagiri, India
- magpi.cc/LCCUjV

A second Jam in Mangalagiri after a successful first one, with beginners welcome.

04. UCF Raspberry Jam

- Monday 15 October
- UCF, Orlando, FL, USA
- magpi.cc/CsgVVr

An event for people of all experience levels to learn about coding and circuit design.

05. Raspberry Jam São Paulo

- Saturday 20 October
- Centro Cultural São Paulo, São Paulo, Brazil
- magpi.cc/oJLWAF

Created to bring together members of the tech community, especially Raspberry Pi enthusiasts.

06. Raspberry Pi Jamwich

- Wednesday 24 October
- The Forum, Norwich, UK
- magpi.cc/sKHCeC

Get your hands on some Raspberry Pi boards during the Norwich Science Festival.

07. Raspberry Jam Manila

- Saturday 27 October
- LaunchGarage, Manila, Philippines
- magpi.cc/AJvjbd

Come together to learn about digital making with Raspberry Pi, presented by #Geeks4Good.

08. Perak Technology Session

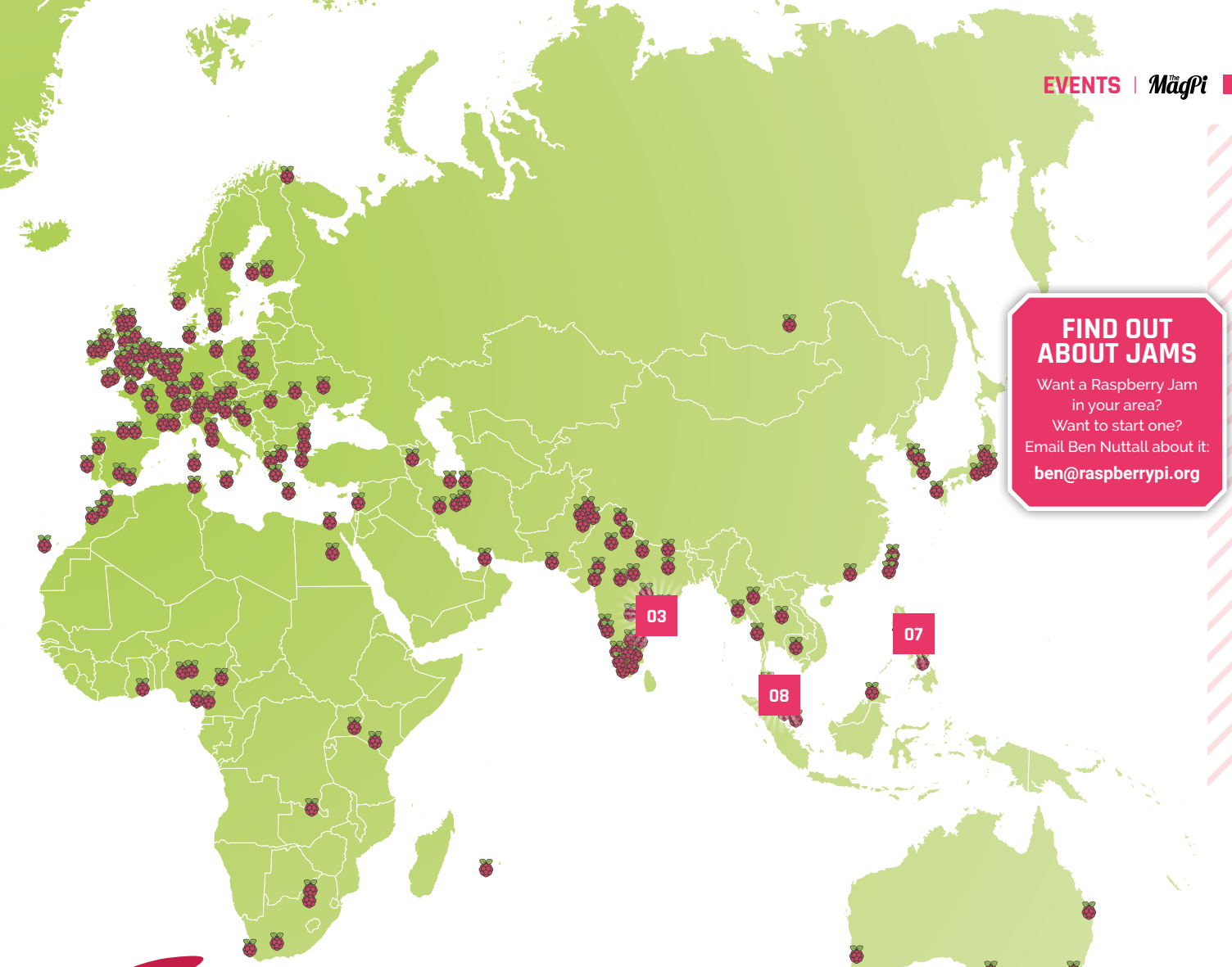
- Saturday 27 October
- HZ Rental Resources, Seri Manjung, Malaysia
- magpi.cc/weyKPJ

This session focusing on the Pi Zero will teach you how to use it and install software.

FULL CALENDAR

Get a full list of upcoming events for October and beyond here:

rpf.io/jam

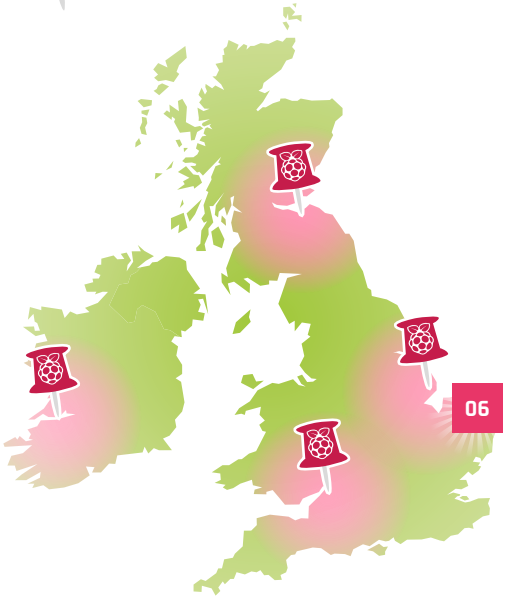


FIND OUT ABOUT JAMS

Want a Raspberry Jam in your area? Want to start one? Email Ben Nuttall about it:

ben@raspberrypi.org

We've highlighted some of the areas in need of a Jam! Can you help out?



Raspberry Jam advice:

Activities

"When I ran the Manchester Jam, people were happy to work on their own projects. Occasionally we ran a track of talks, or put on a workshop, which was a nice change. Now I run the Jam at Pi Towers, the attendees tend to be mostly beginners – usually parents with their children – so we mostly run workshops and hand out worksheets."

Ben Nuttall – Raspberry Jam @ Pi Towers

Every Raspberry Jam is entitled to apply for a Jam starter kit, which includes magazine issues, printed worksheets, stickers, flyers and more. Get the book here: **magpi.cc/2q9DHfQ**

Your Letters

▲ We love inventive Pi projects, like the Aquaponic Garden on page 8

Submitting projects

I've always loved reading *The MagPi* and would love to contribute to it – I've recently completed a project I've been wanting to do for a couple of years. Can I write something about it in *The MagPi*? What is the submission process like?

Ed via email

Submitting a project to *The MagPi* is easy – all you need to do is email us!

Depending on the kind of project, there are several things we can do. Some projects are quite unique and specialised, and we like to do a project showcase on those, with lots of pictures to show off the amazing thing you've made. For some simpler projects, we like to turn it into a guide, showing folks how they can make their own version of the project.

We welcome all projects in the magazine!

Highlighted

I am new to Raspberry Pi. I somehow stumbled on this and I really liked it, so I recently purchased the Raspberry Pi 3B+ kit. I have some programming background but am new to Raspberry Pi. So my question could be weird.

Which text editor is used in *The MagPi* to show Python scripts? I really like the syntax highlighting in it. I tried many (IDLE, gVim, Thonny, Notepad++) but none gives the same results.

I know it is the code which really matters and not the text editor, but I really loved this colour combo.

wingsoffire from the Forum

We actually colour the code by hand in *The MagPi* – you'll see as part of this redesign that they've been updated since you saw a particular bit of code.

A lot of text editors will let you change the code highlighting colour if you go to the preferences – we suggest playing around with them and try to get it either close to how we display it, or tweak it further to create your own perfect personal highlighting scheme.

Seventy-five

As I put my last issue of *The MagPi* on my bookshelf, I noticed that you were close to another milestone in issue 75. Are you planning on doing anything special for it? Should I be telling my friends to get a subscription sorted in case you run out of copies again?

Kieron on Facebook

You should be telling your friends to pick up a subscription to their favourite magazine anyway. We don't pre-announce freebies because we want to ensure our regular readers get them.

We are doing something special, though! Much like issue 50, we're going to do a big countdown of projects. We want the community to vote on the projects again, so please head to **magpi.cc/75projects** and choose your favourite!

▼ Find out the top 75 Raspberry Pi projects in *The MagPi* 75

Contact us!

- Twitter **@TheMagPi**
- Facebook **magpi.cc/facebook**
- Email **magpi@raspberrypi.org**
- Online **raspberrypi.org/forums**

More Astro Pi?

I've been looking out for any information about future Astro Pi projects – I didn't actually know it existed until recently and would love to give it a go! Getting my code into space sounds amazing, and I'm a big proponent of science as well!

Is there any news on what it will be next time, and when?

Lauren via email

You're in luck, Lauren: the next Astro Pi competition has just recently been announced! You can find more info about it here: **magpi.cc/otWxzF**.

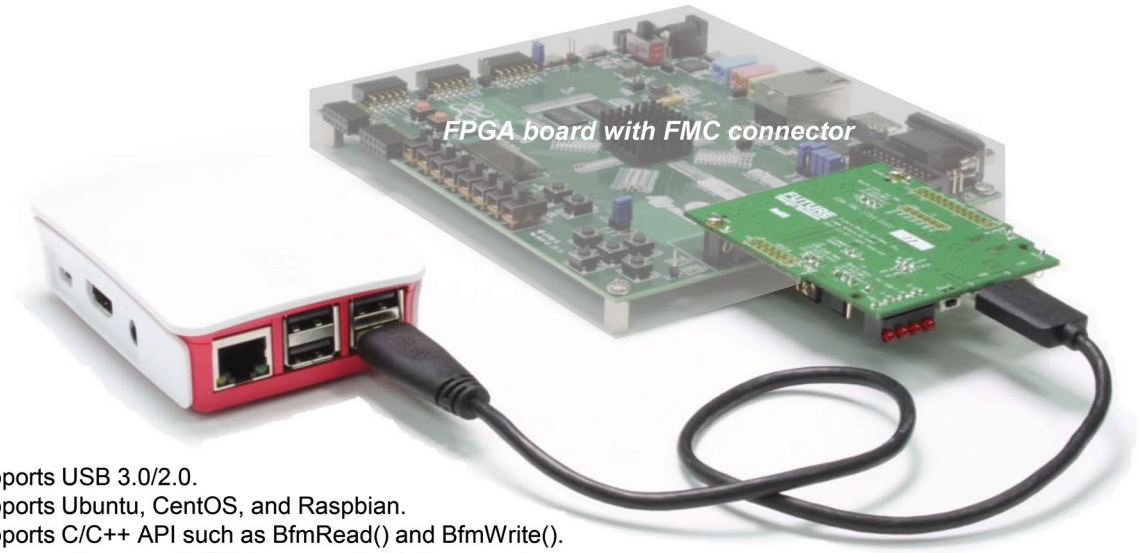
The new competition, Mission Space Lab, works just like the previous competitions; if you're under 19 and live in one of the ESA member state countries, you can form a team and suggest an experiment.

The themes for this competition are Life in Space and Life on Earth – the latter's experiments are able to use Izzy, the Raspberry Pi that looks out of a porthole on the ISS and views the Earth with an infrared camera.

▲ Will you study life in space or on Earth?

There's also Mission Zero for even younger coders, which you can check out the details for on the blog.

Connecting *Raspberry Pi* to *FPGA* through *USB*



- CON-FMC™ supports USB 3.0/2.0.
- CON-FMC™ supports Ubuntu, CentOS, and Raspbian.
- CON-FMC™ supports C/C++ API such as BfmRead() and BfmWrite().
- CON-FMC™ supports off-the-shelf FPGA board with FMC connector.
- CON-FMC™ supports FIFO interface and AMBA AXI/AHB/APB buses.
- CON-FMC™ supports CPU-to-FPGA offloading.

For additional information contact at **contact@future-ds.com** or visit **www.future-ds.com/CON-FMC**

FUTURE Design Systems

WIN One of Five Pi Switch Caps

with an OLED display and weather pack!

> This Pi add-on includes power switch, OLED display, infrared, LED, and I²C ports for weather sensors. It also works as Volumio and Kodi controller

We thought the Pi Switch Cap was the perfect on-and-off switch for your Raspberry Pi when we reviewed it back in issue 70 of ***The MagPi***, especially as you can use it help control Kodi and other HTPC software as well!

Nanomesher has five to give away, so enter today for your chance to win one.

Enter **now** at: magpi.cc/win

In association with

NANOMESHER

Learn more: magpi.cc/cwNDmz

Terms & Conditions

Competition opens on **26 September 2018** and closes on **26 October 2018**. Prize is offered to participants worldwide aged 13 or over, except employees of the Raspberry Pi Foundation, the prize supplier, their families or friends. Winners will be notified by email no more than 30 days after the competition closes. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine. We don't like spam: participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered. Winners will be contacted by email to arrange delivery. Any winners who have not responded 60 days after the initial email is sent will have their prize revoked. This promotion is in no way sponsored, endorsed or administered by, or associated with, Instagram or Facebook.

Stay Tuned

THE MAGPI #75 ON SALE 25 OCT

Also

AIY Edge TPU Accelerator
Raspberry Pi Drone
Pygame Zero Pac-Man
Secure Walkie Talkie
Top 10 DAC HATs
And much, much more

DON'T MISS OUT!
magpi.cc/subscribe

TWITTER	@TheMagPi
FACEBOOK	fb.com/MagPiMagazine
EMAIL	magpi@raspberrypi.org

EDITORIAL
Editor
Lucy Hattersley
lucy@raspberrypi.org

Features Editor
Rob Zwetsloot
rob.zwetsloot@raspberrypi.org

Sub Editors
Phil King and Jem Roberts

DESIGN
criticalmedia.co.uk

Head of Design
Dougal Matthews

Designers
Mike Kay and Lee Allen

Illustrator
Sam Alder

CONTRIBUTORS
Brian Beuken, Mike Cook, David Crookes, PJ Evans, Rosie Hattersley, Nicola King, Jon McLoone, Liz Upton, Mark Vanstone

PUBLISHING
Publishing Director
Russell Barnes
russell@raspberrypi.org
+44 (0)7904 766523

Director of Communications
Liz Upton

CEO
Eben Upton

DISTRIBUTION
Seymour Distribution Ltd
2 East Poultry Ave,
London EC1A 9PT
+44 (0)207 429 4000

SUBSCRIPTIONS
Raspberry Pi Press
Mann Enterprises, Unit E,
Brocks Business Centre,
Haverhill, CB9 8QP

To subscribe
magpi.cc/subscribe

To get help:
magpi.cc/subshelp

recycle

BY NC SA

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

The MagPi magazine is published by Raspberry Pi (Trading) Ltd., 30 Station Road, Cambridge, CB1 2JH. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products or services referred to or advertised in the magazine. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0).
ISSN: 2051-9982.

Living the **Pi life**

Liz Upton on a decade of solder burns and paper cuts

I'm married to Raspberry Pi in more than one way. Eben Upton and I have been together for 20 years now. We met at university, where he wooed me with an interpreted Java spinning torus, with a creepy clown mapped onto it. We talked about our BBC Micro-infused childhoods: as a kid I subscribed to Micro User magazine for the type-in listings, typeset my own monthly family newspaper, built databases about the things I found in Smash Hits magazine, played all the games I could get my hands on, and embarked on a doomed attempt to build my own 'Hitchhiker's Guide to the Galaxy'; while a tiny Eben was busy writing mouse drivers and doing impenetrable stuff with machine code at the other end of the country.

> I dropped all my freelance work the very next day and started volunteering full-time for Raspberry Pi

Something had changed

The conversation that became Raspberry Pi started at university, and has never really stopped. Where were the resources, machines, and groups of people we'd learned from as kids? Neither of us learned any serious computing at school. Magazines, groups of friends, informal computer clubs and, in my case, the very kind father of a friend (who got me set up with the typesetting and database software I had so much fun with, because I was a weird kid) were part of it – but where were the platforms that kids could learn on? Even back then, during the dot-com boom, it was apparent that something had changed, with games consoles and the precious family PC replacing the old, hackable machines that we used to use.

We noodled around the topic for years, and started to formalise a plan with some friends in Cambridge around 2009. Things didn't move very fast; we were busy with other things (in my case, the building of a successful career as a freelance journalist and editor; I'd just worked on a book with National Geographic and won one of the beauty industry's top awards for writing). Then, in 2011, a video Rory Cellan-Jones, the BBC's Chief Technology Correspondent, made about our plans for a little, programmable device that kids could buy with pocket money went viral. We spent the evening feeling pleased with ourselves and then realised with a sudden horror that we'd just promised the 600 000 people who'd just watched it that we'd make them a computer for $25.

Full-time job

I dropped all my freelance work the very next day and started volunteering full-time for Raspberry Pi. I built our first website (lousy, but it did the job), took on responsibility for press and social media, worked on building a community, filled envelopes with the stickers we sold to raise a little initial capital, wrote articles and blog posts, learned to solder, and got a lot of paper cuts. I didn't actually get around to going on the payroll until about a year after we'd started paying the first members of staff (some months after we'd actually launched the first physical Raspberry Pi boards) – it didn't feel like a job. It felt like my whole life.

And it still does. It's been a great ten years...

AUTHOR

Liz Upton

Liz is Raspberry Pi's Executive Director of Communications. She restores fountain pens, plays the piano, and enjoys her toddler daughter, just not all at the same time.

magpi.cc/liz

HackSpace

TECHNOLOGY IN YOUR HANDS

THE NEW MAGAZINE FOR THE MODERN MAKER

SUBSCRIBE AND SAVE UP TO 35% on the cover price

ISSUE #11

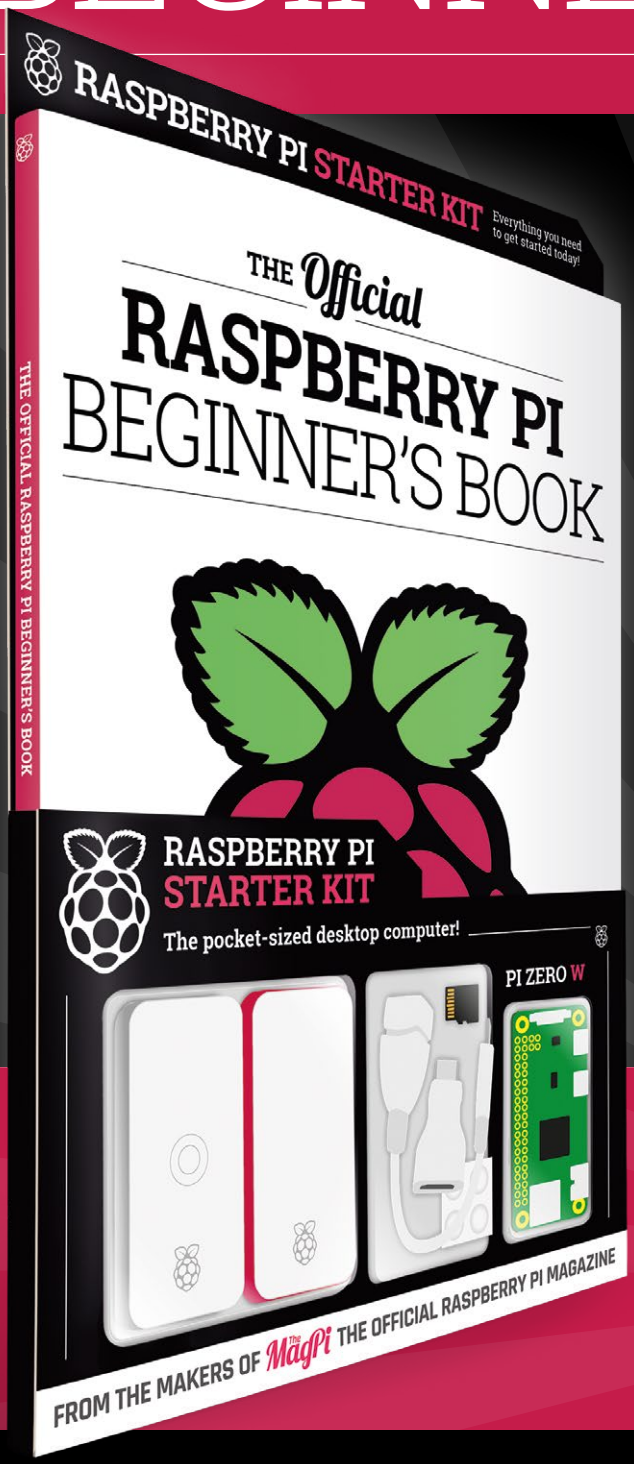
OUT NOW

hsmag.cc

Available on the App Store

GET IT ON Google Play

THE *Official* RASPBERRY PI BEGINNER'S BOOK


RASPBERRY PI STARTER KIT
Everything you need to get started today!
THE Official RASPBERRY PI BEGINNER'S BOOK
THE OFFICIAL RASPBERRY PI BEGINNER'S BOOK
RASPBERRY PI STARTER KIT
The pocket-sized desktop computer!
PI ZERO W
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE

LEARN COMPUTING THE EASY WAY!

Includes

- Pi Zero W computer
- Official case with three covers
- USB and HDMI adapters
- 8GB microSD card
- 116-page beginner's book

Available now

Buy online: magpi.cc/store